corr. US 5,583,500

[19]中华人民共和国专利局

[11] 公开号 CN 1124889A

[12] 发明专利申请公开说明书

[21]申请号 94120457.X

[51]Int.Cl[6]

H03M 7/30

[43]公开日 1996 年 6 月 19 日

[22]申请日 94.12.22

[30]优先权

[32]93.12.23[33]US[31]172646

[71]申请人 株式会社理光

地址 日本东京都

[72]发明人 J·D·阿伦 M·波力克

M·戈米什 E·L·许瓦茨

[74]专利代理机构 中国专利代理(香港)有限公司

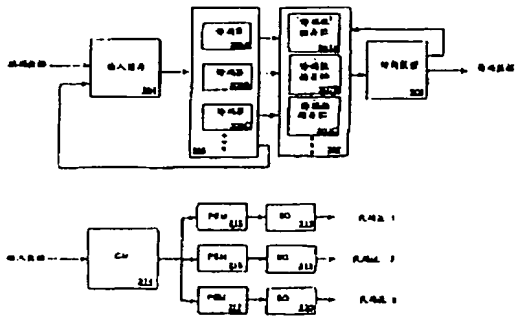
代理人 杜有文 马铁良

权利要求书 12 页 说明书 65 页 附图页数 26 页

[54]发明名称 数据并行编码和译码的方法和装置

[57]摘要

本发明提出了一种并行数据编码和译码的方法和装置。本发明提出对具有多个代码字的数据流进行去压缩（decopmpressing）的系统。此系统包括有一接收数据流的输入信道。此系统还包括有一对数据流的每一比特位进行译码的译码器，其中，数据流中至少两个代码字被同时加以译码，从而使数据流被并列译码。



权 利 要 求 书

1.对多个代码字的数据流进行译码的方法，其特征是具有下列步骤：

将数据流的部分分派给多个译码设备；

利用该多个译码设备对该数据流的部分进行译码，其中至少该多个译码设备中之一个在一流水线方式的连续周期内对其相应的数据流部分进行译码。

2.权利要求1所述的方法，其特征是所述译码步骤包括有以流水线方式对数据流部分进行译码的步骤。

3.权利要求1所述的方法，其特征是还进一步包括将数据流排列成一被排序的数据流。

4.权利要求3所述的方法，其特征是所述分派步骤包括按照数据流的顺序来分派这些部分。

5.权利要求1所述的方法，其特征是其中多个译码设备包括许多QM编码器。

6.权利要求1所述的方法，其特征是所述多个译码设备包括许多有损译码器。

7.权利要求1所述的方法，其特征是所述多个译码设备包括第一类型译码器和第二类型译码器，其中第一类型译码器与第二类型译码器不同。

8.权利要求7所述的方法，其特征是所述第一类型译码器包含一行程编码器，所述第二类型译码器包含一Q编码器。

9.权利要求7所述的方法，其特征是所述第一类型译码器包含一二进制译码器，所述第二类型译码器包含一非二进制译码器。

10.权利要求7所述的方法，其特征是所述第一类型译码器包含一无损译码器，所述第二类型译码器包含一有损译码器。

11.对多个代码字的数据流进行译码的方法，其特征是包括下列步骤：

确定当前的结构槽；

由存贮器检索该结构槽的译码器状态；和

对一代码字进行处理，包括对所述一代码字进行译码的步骤，其中确定、取数和处理的步骤在流水线方式中的连续周期中进行。

12.权利要求11所述的方法，其特征在是还进一步包括有使数据流作向跟随所述一代码字之后的代码字作移位的步骤。

13.权利要求11的方法，其特征是所述取该译码器状态的步骤包含有步骤：

取PEM状态；和

取位发生器状态。

14.权利要求11所述的方法，其特征是所述处理步骤包含有步骤：

产生一毕特位；和

对所述一代码字进行译码，使得在进行所述一代码字的译码之前产生该毕特位。

15.权利要求14所述的方法，其特征是所述产生和译码步骤在相连续的周期内进行。

16.权利要求11所述的方法，其特征是所述处理步骤包含有步

骤：

对所述一代码字进行译码，并使得在所述一代码字的译码之先产生该毕特位；和

产生一毕特位。

17.权利要求11所述的方法，其特征是进一步包括对PEM状态进行译码以确定为对所述多个代码字之一进行译码所需的一代码的步骤。

18.权利要求11所述的方法，其特征是进一步包括更新PEM状态以在所述多个代码字之一已被译码后产生一被更新的PEM状态的步骤。

19.权利要求18所述的方法，其特征是所述更新和译码步骤在同一周期内进行。

20.权利要求18所述的方法，其特征是还进一步包含将被更新的PEM状态写进存贮器的步骤。

21.对多个代码字的数据流进行译码的方法，其特征是包括有步骤：

确定针对所述多个代码字之一的当前结构槽；

取出针对该当前结构槽的概率估算机(PEM)的状态；

取出针对当前结构槽的位发生器状态；

产生表示正进行译码的所述多个代码字之一的第一毕特位的毕特位；

对PEM状态进行译码以确定一针对所述多个代码字之一的译码的一代码；和

利用该代码对所述多个代码字之一进行译码。

22.权利要求21所述的方法，其特征是所述产生一毕特位和对所述多个代码字之一进行译码的步骤在一流水线状态中的相连续周期中进行。

23.权利要求21所述的方法，其特征是所述确定，取数和产生步骤以一流水线方式在连续的周期中进行。

24.权利要求23所述的方法，其特征是所述产生一毕特位和对PEM状态译码的步骤在一流水线方式的相连续周期中进行。

25.权利要求21所述的方法，其特征是还进一步包括对紧随所述多个代码字之一的后面的代码字再利用所述当前结构槽的步骤，在此，免除了访问外存贮器以获得结构信息的需要。

26.用于去压缩具有多个代码字的数据流的系统，其特征是包括有：

用于接收数据流的多个代码字的信道装置；和

用于与所述信道装置相耦合以对多个代码字中的每一个进行译码的译码器装置，其中代码字以流水线方式进行译码。

27.权利要求26所述的系统，其特征是其中数据流的至少两个代码字同时被加以译码，从而使数据流被并列进行译码。

28.权利要求26所述的系统，其特征是所述译码器装置包括有多个译码器。

29.用于对多个代码字的数据流进行去压缩的系统，其特征是包括：

用于接收数据流的多个代码字的信道控制装置；

多个连接来由该信道控制装置接收多个代码定的位流发生器，其中多个位流发生器中的每一个利用至少一个代码来对每一代码

字进行译码，所述至少一代码用于对固定长的代码字进行译码，同时其中多个位流发生器的至少两个同时对代码字进行译码，从而使数据流被并列译码产生译码数据；和

模拟装置，连接到多个位流发生器用于由多个位流发生器选择译码数据以输出该译码数据。

30. 权利要求29 所述的系统，其特征是所述至少一码为一Tunstall码。

31.权利要求29所述的系统，其特征是每一代码字含有n毕特位，其中n为8的倍数。

32.将多个代码字的数据流加以去压缩的方法，其特征是包括步骤：

接收数据流的多个代码字；

利用一Tunstall码对每一代码字进行译码，所述Tunstall码用于对固定长度的代码字进行译码，而且其中多个位流发生器的至少两个同时对代码字译码，以使得对数据流作并行译码来产生译码数据；和

由多个位流发生器选择译码数据加以输出。

33.去压缩多个代码字的数据流的系统，其特征是包括：

接收数据流的多个代码字的信道控制装置；

多个被连接来由信道控制装置接收多个代码字的位流发生器，其中多个位流发生器中的每一个利用至少一个非二进制代码来对每一代码字译码，而且多个位流发生器的至少两个同时对代码字进行译码，以便对数据流作并行译码产生译码数据；和

模拟装置，连接到多个位流发生器用于由此多个位流发生器选

择译码数据加以输出。

34.权利要求33所述的系统，其特征是所述至少一代码字包括一整个值。

35.权利要求33的所述系统，其特征是所述多个位流发生器利用多个非二进制编码。

36.去压缩具有多个代码字的数据流的系统，其特征是包括有：

用于接收多个代码字的信道装置；

与信道装置相连的多个毕特位发生器，用于由信道装置连续地接收多个代码字并对之进行译码以产生被译码的数据流；和

与多个毕特位发生器相连并对之进行控制的控制装置，其中多个毕特位发生器的每一个由信道装置接收一代码字以进行译码，以使得至少在多个毕特位发生器中的两个对连续的代码字进行译码。

37.权利要求36所述的系统，其特征是所述多个位发生器的每一个对数据流中第n个代码字译码，其中n为此多个位流发生器的数目。

38.权利要求36所述的系统，其特征是所述多个位发生器包括多个霍夫曼译码器。

39.权利要求36所述的系统，其特征是所述控制装置包括一状态机。

40.权利要求36所述的系统，其特征是所述多个位流发生器由多个霍夫曼译码器组成，所述信道装置包括有一缓存器，其中所述控制装置将数据顺序提供给此多个霍夫曼译码器中的每一个，以使每一霍夫曼译码器接受第n个标记，n为霍夫曼译码器数。

41.权利要求40所述的系统，其特征是代码字被传送给多个霍夫曼译码器，对标记以与编码期间的同样顺序进行译码。

42.权利要求40所述的系统，其特征是所述多个霍夫曼译码器

按照JPEG标准进行译码，除开是以代码字的交错处理顺序以外。

43.权利要求40所述的系统，其特征是所述多个霍夫曼译码器按照NPEG标准进行译码，除开是以代码字的交错处理顺序以外。

44.权利要求40所述的系统，其特征是所述多个霍夫曼译码器的至少一个利用多个可行的霍夫曼编码之一对数据进行译码，在此，多个可行的霍夫曼编码的每一个根据当前值的概率来加以选择。

45.权利要求36所述的系统，其特征是所述多个霍夫曼译码器的至少一个根据码元的自适应概率为一特定码元转换相应的代码字。

46.权利要求40所述的系统，其特征是所述控制装置包括一状态机。

47.权利要求36所述的系统，其特征是所述控制装置包括一结构模型。

48.权利要求36所述的系统，其特征是还进一步包括有至少一个JPEM QM编码器。

49.权利要求40所述的系统，其特征是还进一步包括一用于确定二元判定结果的QM编码器，在此，所述多个霍夫曼译码器按该结果指定系数值。

50.由多个代码流组成一代码流的方法，其中多个代码流的每一个均包括有多个代码字，所述方法包括步骤：

将多个代码流的每一个中的代码字装配成固定长的字；

按照该固定长字中代码字的被分派的序列将多个代码流中的代码字进行交错排列，以使得组成代码流包括多个代码流中的代码字。

51.权利要求50所述的方法，其特征是所述交错排列步骤包括按照最接近组成代码流的起始的代码字作交错处理。

52.权利要求50所述的方法，其特征是进一步包括有按照模拟顺序分派多个代码流中的代码字。

53 .权利要求50所述的方法，其特征是所述多个代码流的一个包括音频数据而另一所述多个代码流包括视频数据，从而使组合数据流包括有交错排列的声频数据和视频数据。

54.对具有多个代码字的代码流进行译码的系统，其特征是包括有：

一由代码流提供代码字的缓存器；

一提供结构的结构模型；

一连接到结构模型用于存放状态信息的存贮器，此存贮器按照结构模型提供的每一结构给出状态信息；和

多个连接到缓存器和该存贮器的多个译码器，用于利用由存贮器得到的状态信息对缓存器所给出的代码字进行译码，其中此多个译码器的两个对代码字作并行译码，以使得此多个译码器对数据流进行译码。

55.权利要求54所述的系统，其特征是所述存贮器与所述多个译码器及所述结构模型一齐被置于一集成电路上。

56.权利要求54所述的系统，其特征是所述存贮器为片载存贮器，用于存放公用结构的信息。

57.权利要求54所述的系统，其特征是所述存贮器为片载存贮器，存放较大运行计数的最低有效位部分，并在运行的末尾访问外存贮器。

58.权利要求54所述的系统，其特征是所述存贮器为片载存贮器，存放位发生器一部分的状态信息，并对其余状态信息采用外存贮器。

59.权利要求54所述的系统，其特征是所述存贮器包括至少两个存贮器分区，其中所述两个存贮器分区为分开的结构槽组存放状态信息，而且其中所述至少两个存贮器分区以交替的周期进行访问。

60.对多个代码字的图象数据流进行JPEG去压缩的方法，其特征是包括步骤：

将数据流的部分分派给多个译码设备；

利用此多个译码设备对数据流这些部分进行译码，其中该多个译码设备的至少两个包括R编码器。

61.权利要求60所述的方法，其特征是所述译码步骤包括利用无损二元熵编码器对该数据流的部分进行译码。

62.对数据流进行编码的方法，其特征是包括步骤：

以有限量的缓冲存贮器进行编码；和

在一编码器和一译码器间作信号传输来指出已到达一存贮器极限。

63.权利要求62所述的方法，其特征是所述编码器响应指出已到达存贮器极限的所述信号产生一输出。

64.权利要求62所述的方法，其特征是作信号传输的步骤包括加入后面未跟随有一最小或然码元的最大或然码元的非最大行程。

65.权利要求62所述的方法，其特征是所述信号传输步骤包括有步骤：

将数据流中的代码字作时间标记；

在作时间标记后缓存代码字；和

在达到存贮器极限时以最低时间标记完成代码字。

66.权利要求62所述的方法，其特征是所述信号传输步骤包括有利用分隔的结构槽将存贮器的状态送给译码器的步骤，其中结构槽被对每一代码字加以译码以指明该代码字是否取决于缓存溢出。

67.一编码器，其特征是包括：

一用于接收多个代码字的数据流的序列器；

连接到该序列器用于对多个代码字进行编码的编码器；

连接到此多个编码器、或对该编码的代码字加以缓存的缓存装置；

用于由该缓存装置求取被编码数据以产生交错排列的译码数据的输出数据流的交错处理机构；和

与该交错处理机构相连的对编码数据时的顺序进行模拟的译码器，其中此译码器以流水线方式运行。

68.权利要求67所述的编码器，其特征是还包括有指出该缓存装置发生溢出的装置。

69.权利要求68所述的编码器，其特征是所述指示装置包括至少一个予定的代码字。

70.一编码器，其特征是包括：

一接收多个代码字的数据流的序列器；

与该序列器相连、对多个代码字进行编码的多个编码器；

连接到此多个编码器、 或对被编码代码字进行缓存的缓存装置；

由缓存装置求取被编码数据以产生交错排列的编码数据的输

出数据流的交错处理机构，其中交错处理机构包括有输出在缓存装置的存贮器到达其极限时未被完全编码的数据的装置。

71.权利要求70所述的编码器，其特征是还包括有 加入未跟随最小或然码元的最大或然码元的非最大行程的装置。

72.权利要求70所述的编码器，其特征是还包括有：

对数据流中代码字作时间标记的装置；

连接到时间标记装置对经过加以时间标记后的代码字进行缓存的装置；和

在存贮器达到其极限时以最低时间标记完成代码字的装置。

73.对多个代码字的数据流进行去压缩的实时视频系统，其特征是包括：

接收该数据流以对数据流中每一代码字进行译码的第一多个译码设备，在此，数据流以流水线方式进行译码；

对被码数据流给定格式以为将该被译码数据流加以显示作准备的格式化装置；和

显示被译码数据的显示装置。

74.权利要求73所述的系统，其特征是所述显示装置为一电视监视器。

75.权利要求73所述的系统，其特征是所述显示装置为一视频监视器。

76.权利要求73所述的系统，其特征是第二译码器装置为一有损译码器。

77.权利要求76所述的系统，其特征是所述有损译码器包括一译码器的变换部分。

78.权利要求76所述的系统，其特征是所述有损译码器包括一种译码器的彩色变换部分。

79.权利要求76所述的系统，其特征是所述有损译码器包括一译码器的二次采样部分。

说 明 书

数据并行编码和译码的方法和装置

本发明是关于数据压缩(Compression)和去压缩(decompression)系统技术领域，特别是关于在压缩/ 去压缩系统中的数据的并行编码和译码方法和装置。

本申请是名为"数据的并行译码和编码的方法和装置" 的美国专利申请(No 08/ 016 035,Feb.10,1993)的延续部分。

今天，广泛采用数据压缩技术，特别是在存贮和传送大量的数据场合。现有技术中有许多不同的压缩技术。 这些压缩技术可被分成两大类:有损编码和无损编码。有损编码是指那些会导致信息丢失的编码，因而不能保证完全地再现原始数据。而无损压缩中，所有信息均被保留着，对数据的压缩过程是使得数据能完全复元的。

在无损压缩中，输入信号码元被变换成输出代码字。如果压缩是成功的，表示这些代码字的信息位(毕特)数量就会小于输入码元的数量。无损编码方法包括词典编码法(例如，Lempel-Ziv)，行程扫描宽度持续长度编码法，枚举编码法和熵编码法。

熵编码是指任一设法利用已知的或估算的码元概率将数据压缩到接近熵(平均信息量)的极限的无损编码法。 熵编码包括霍夫曼编码，算术编码和二元熵编码。二元编码器是仅针对二元(是/非)判定工作的无损编码器，此二元判定经常表示为最大或然码元(MPS)和最小或然码元(LPS)。作为二元熵编码器例子的有IBM的 Q

编码器和一种称之为B编码器的编码器。关于B 编码器的详细情况可参看美国专利No 5272478 J.D.Allen的"熵编码的方法和装置"(Dec.21,1993) 转让于本发明的共同受让人。 也可参看 M. J- . Gormish & J. D. Allen" 有限状态机的二元熵编码 "(Proc. DataCompression Conference 中的摘要(March 30, 1993, Snowbird,UT,P.449)。B编码机是一种用有限状态机压缩的二元熵编码器。

图1所示为一现有技术采用二元熵编码器的压缩和去压缩系统的方法图。进行编码时,数据输入进(前后关系)结构模型(CM)101。CM101将输入数据变换成一组(或序列)二元判定并给每一判定提供结构槽(Conetext bin)。 此判定序列及它们的相关结构槽一同由CM101输出到概率估算组件(PEM)102。PEM102接收每一结构槽并对每一二元判定产生一概率估算。 实际的概率估算通常均以一称之为P类别的类别(class)来表示。每一 P 类别用于一个概率范围。PEM102还确定二元判定(结果)是否是处于它的较大可能的状态(即是否该判定对应于MPS)。位流发生器(BG)组件103接收此概率估算(即P类别)及关于此二元判定是否可能是输入的确定。据此,BG 组件103产生压缩数据流,输出零或多个毕特位以代表原始数据。

译码时,CM104提供结构槽到PEM105,而PEM105即根据此结构槽给出概率类别(P类别)到BG组件106。BG106被连接来接收概率类别。根据此概率类别及压缩数据,BG组件106回复一表明该二元判定(即事件)是否处于它的最大或然状态的毕特位。PEM105接收此毕特位,根据此接收到的毕特位更新概率估算,并将结果送回CM104。CM104接收此回复毕特位并用它来产生原始数据并为下一二元判定更新

结构槽。

采用诸如IBM的Q编码器和B编码器等的二元熵编码器的一个问题是它们很缓慢，即使用硬件实现也如此。它们的运行需要一个单独的很慢的大反馈电路。为重述译码过程，结构模型利用过去被译码的数据来产生一结构。概率估算组件利用此结构产生一概率类别。位流产生器利用概率类别及压缩数据以确定下一毕特位可能是或者不可能是结果。概率估算组件利用此可能/不可能的结果产生一结果位(以及针对该结构更新概率估算)。此结果位被结构模型用来更新它的已往数据的过程。为对一单个的毕特位进行译码均需要全部这些步骤。因为结构模型在能提供下一结构前必须等待该结果位来更新它的历史过程，所以对下一毕特位的译码就必须等待。这就希望能避免在对下一毕特位译码前必须完成反馈电路的操作。换句话说，希望能一次对多于一毕特位或代码字进行译码以便增加对压缩数据译码的速度。

利用二元熵编码的译码器的另一个问题是必须处理可变长数据。在大多数系统中，欲加以译码的代码字均具有可变长度。另外其他一些系统则对可变长度的码元(未编码数据)进行编码。在处理可变长数据时，需要以位为单元移动数据以便能提供译码或编码运行中正确的下一数据。这些以位为单位对数据流的处理可能需要昂贵的和/或缓慢的硬件和软件。而且，现有技术的系统要求在时间要求苛刻的反馈电路中完成这种移位，这就限制了译码器的性能。如能对于编码和译码操作中仅有一个昂贵和/或速度受限的情况允许这一对数据流作位为单位的处理或者在编码器中进行或者在译码器中进行将是很有利的。同样也有利的是去除时间受限

的反馈环路中的这种以位为单位对数据流的处理过程，从而能采采并行操作来提高速度。

本发明提出一种无损压缩和去压缩系统。本发明提出一种能快速译码的简单的译码器。本发明还提供以并行和流水线方式对数据进行译码的译码系统。本发明提出的对二元熵编码数据流进行的译码操作，无需执行现有技术中的在苛刻的反馈电路中的以位为单元的处理。本发明还提出多个编码器（无损或有损）数据的交错应用而无需采用过多的标志。

本发明提出的数据去压缩和压缩的方法和装置中，包括一种对具有多个代码字的数据流以流水线方式进行译码的方法和装置。本发明包括有将数据流的部分分派到多个译码设备的方法和装置。这些译码设备对数据译码，其中至少一个译码设备对其对应的数据流部分在连续的循环周期内以流水线方式进行译码。

在另一实施例中，对数据流的译码从确定当前的结构槽开始。当前的结构槽的译码器状态由存贮器提取得。由此就可能确定概率估算机（PEM）的状态。然后对代码字进行处理并译码。在本发明中，结构的确定、PEM 状态的确定和代码字的处理是在流水线方式的连续周期内进行的。

本发明还提出一种利用可变长数据的并行译码和编码的方法和装置。欲译码的数据可以是从有损或无损编码器接收的。同样，接收的数据是经过编码的或未经过编码的。

由以下的详细说明和发明实施例附图将会更全面理解本发明，但这些实施例仅用于说明和了解，而不是对本发明的限制。

所列附图分别为：

图1为现有技术中二元熵编码器和译码器的方框图；

图2A为本发明的译码系统的方框图；

图2B为本发明编码系统一实施例的方框图；

图2C为并列处理结构槽的本发明译码系统一实施例的方框图；

图2D 为并列处理概率类别的本发明译码系统一实施例的方框图；

图3说明本发明的无交错编码流；

图4说明由一典型数据组推导得的交错编码流的一实施例；

图5为本发明的R 编码器中的概率估算表和位流发生器的一个示例；

图6为本发明的译码系统一实施例的方框图；

图7为本发明的译码器一实施例的方框图；

图8说明本发明的译码流水线的一实施例；

图9说明本发明所采用的结构模型的模板的一举例；

图10为本发明的位发生电路一实施例的方框图；

图11说明本发明中对PEM状态译码用的逻辑；

图12 为用于确定是否需要一新代码字的逻辑的一实施例的方框图；

图13为本发明的译码逻辑一实施例的方框图；

图14为更新运行计数的逻辑的一实施例的方框图；

图15为本发明的将编码 数 据 移进译码器的一实施例的方框图；

图16A说明本发明的译码系统的实现方案；

图16B说明本发明的译码系统的另一实现方案；

图17为说明本发明的流水线式译码的时序图；

图18 为作隐含信号控制的有限存贮器编码器的一实施例的方框图；

图19 为具有作隐含信号控制的有限存贮器的译码器的一实施例方框图；

图20为本发明一编码器的实施例方框图；

图21为本发明一译码器的实施例方框图；

图22为二分化的判定树；

图23为本发明的声/ 视频信号交错系统的方框图；

图24为应用本发明的高频宽系统的方框图；

图25为应用本发明的带宽匹配系统的方框图；和

图26为应用本发明的一实时视频系统的方框图。

在下面对一数据并行编码和译码的方法和装置的介绍中，对许多方面作了特定的细节描述，例如特定的毕特位数，译码器数，特定的概率，数据类型等，以便能对本发明的优选实施例作透彻的理解。对本技术领域的熟悉人员来说将会明白无须这些特定的细节也可实现本发明。 而且对相当公知的电路均是以方框图而不是它们的细节表示的，以避免对本发明发生不必要的误解

本发明提出一种对被编码数据作并行无损译码的系统。 数据并行译码是采用多重译码设备。 此多重译码设备的每一个均被分配给欲译码的数据流的一部分。数据流的分派是动态地进行的，其时各译码设备同时进行数据译码，从而实现对数据流的并行译码。为使得数据分配能有效地利用这些译码设备，将数据流进行了排序。这被称之为数据流并行化。 数据使得排序每一译码设备可对任何

或者全部编码数据进行译码而无需要等待由结构模型的反馈。

图2A说明本发明的译码系统，不具有现有技术的缓慢的反馈电路。输入缓存204接收编码数据(即代码字)和来自译码器205 的反馈信号，并将编码数据以预定的顺序(例如结构槽顺序)送至本发明的对编码数据译码的译码器205。译码器205包含多重译码器(例如，205A，205B，205C等)。

在一实施例中，每一译码器205A-205C 被施加以针对一组结构的数据。译码器205中的每一译码器由输入缓存204 针对其结构组中的每一结构槽供给编码的数据。利用这种数据，各译码器205A、205B、205C等产生针对其结构槽组的译码数据。 为将被编码数据与一特定结构槽组相关连并不需要该结构模型。

经译码的数据由该译码器205送至译码数据存贮器207(例如，207A，207B，207C等)。应指出的是译码数据存贮器207有可能存放既非经编码的又非未经编码的中间数据，例如运行计数。在这种情况下，译码数据存贮器207以紧致的，但不是熵编码的方式存放数据。

结构模型206独立地运行，根据它送往译码数据存贮器207的反馈信号由译码数据存贮器207(即207A，207B，207C等)接收先前被译码的数据。因此，存在有两个独立的反馈电路，一个处于译码器205和输入缓存204之间，第二个处于结构模型206 和译码数据存贮器207之间。由于省略了大的反馈电路，译码器205 中的译码器(如205A，205B，205C)一旦在由输入缓存204 接收到与它们相关的代码字之后就可立即对之进行译码。

结构模型设置编码系统的存贮器部分并根据此存贮器将一组数据(例如，一幅图象)划分不同的种类(例如，结构槽)。 在本发明

中，结构槽被看作是一些独立的排序的数据组。在一实施例中，各组结构槽具有它们自己概率模型，而每一结构槽具有它们自己的状态(这里概率估算模型被共享)。因此，每一结构槽就有可能采用不同的概率估算模型和/ 或位流发生器。

这样，数据就被加以排序，或并行化，并将数据流的各部分分配给各个编码器予以译码。

为了使数据流并行化，可将数据按照(上下关系)结构、概率或者砖瓦结构来划分。本发明编码系统由经结构模型(CM)区分开的数据输入的并行编码部分如图2B中所示。

参看图2B，此与结构相关的并行编码器部分包含有结构模型(CM)214，概率估算组件(PEM)215-217，和位流发生器(BG)218-220。CM214被用来接收编码输入数据。CM 214还被连接到PEM215-217 。PEM215-217。PEM215-217还分别连接到BG218-220，后者分别输出编码流1、2和3。每一对PEM和BG组成一编码器。因此，此并行编码器表现为三个编码器。虽然这里仅表示出三个并列码，但任何数量的编码器都可以采用。

CM 214以与通常的CM同样的方式将数据流划分为不同的结构，并将此多个数据流送至并行硬件编码设备。各个别的结构，或结构组件针对各自分开的概率估算器(PEM)215- 217 及位发生器(BG)218-219。每一BG218-220输出编码数据流 。

图2C 为本发明的译码系统的译码器部分一实施例的方框图。参看图2C，一与结构相关的并行译码器表示为具有BG221-223，PEM 224-226和CM227。编码流1-3被分别连通到BG221-223。BG 221-223也分别连通到PEM 224-226。PEM224-226 连接到输出再生的输

入数据的CM227。输入来自表示为编码流1-3的数个编码流。一个编码流被分配给各个PEM和BG。每一BG221-223送回一表明二元判定是否处于其较大或然状态的毕特位，PEM224-226用它来返回译码毕特位(如二元判定)。每一PEM224-226与BG221-223之一相关连，指明由输入编码流中该应用哪一编码来产生数据流。CM227以适当的顺序由位流发生器选择被译码的特位来产生经译码的数据流，由此再生原始数据。这样，CM227就由适当的PEM和BG取得去压缩的数据位，实际上将数据重新排序成原始的顺序。应该注意的是，对于这种设计的控制流向是与数据流的方向相反的。BG和PEM有可能在CM227需要以前就对数据译码，而在其前面留有一个或多个毕特位。换句话说，CM227可能要求来自一个BG和PEM(但未接收)的一个毕特位，然后在利用此先前要求的毕特位之前又要求来自的一个或更多的毕特位。

图2C中所示的结构是设计得使PEM与BG紧密地耦合。IBM Q 编码器是具有紧密耦合的PEM与BG的编码器的典型例子。它们之间的局部反馈电路并不成为系统性能的根本制约。

在一不同设计中，所连接的PEM区分数据并将其送到并行BG 单元。这样，就将仅有一个CM和PEM，而BG则被重复。在这种情况下可以利用自适应霍夫曼编码及有限状态机编码。

在图2D中列出了采用PEM来区分数据并将其送至并行BG的类似的译码系统。在这种情况中，概率类别作并行处理，每一位流发生器被指派给一特定的概率类别并接收结果信号。参看图2D，编码数据流1- 3 被连通到用来接收它们的多路位流发生器(例如BG232，BG233，BG234等)。每一位流发生器均连接到PEM235。PEM235 也连

接到CM236。在此结构中，每一位流发生器对编码数据进行译码，其译码结果由PEM235(而不是由CM236)加以选择。每一位流发生器从与一概率类别相关的设备接收编码数据(亦即，此时编码数据可能来自任一结构槽)。PEM235利用概率类别选择位流发生器。概率类别由为CM236所提供给它的结构槽来指明。在这种情况下，译码数据由并行处理概率类别来产生。

本发明的并行译码系统有多种实现方案。在一实施例中，对应于多重结构槽的编码数据流可按各不同编码器的要求被交错排列进一个数据流。在本发明的当前优选实施例中，编码数被排列得即使编码数据是在一个数据流中传送给译码器时每一编码器均能不断地有数据供给。应指出，本发明可对所有类型数据操作，包括图象数据在内。

依靠利用那些能在集成电路中廉价地重复实现的小型简单的译码器，编码数据可被并行地快速译码。在一实施例中，译码器采用现场可编程门阵列(FPGA)芯片或标准单元的专用集成电路(ASIC)芯片硬件实现。并列化与简单的位流发生器的组合使得对编码数据的译码过程能以超过现有技术译码器的速度进行，但同时保持现有技术译码系统的压缩效率。

存在许多影响系统性能的各种不同的设计上的争论和问题。下面将谈到它们中的几个。不过图2B和2C(及2D)中所示的设计应用多路编码流。具有能协同这一设计的并行信道的系统可以想象为：多路电话线，多个磁盘驱动器上的磁头等等。在一些应用中，仅仅只可能(或方便)采用一个信道。实际上，如果要求用多重信道时可能恶化带宽的应用，因为个别编码流具有猝发特性。

在一实施例中，编码流被加以衔接起来，连续地送往译码器。一前导标题包含有指向每一数据流的起始位存放单元的指针。图3说明这种数据安排的一实施例。参看图3，三个指针301-303分别指出的编码流1、2和3的连接编码中的起始位置。一完整的压缩数据文件可由译码器的缓存中得到。在需要时，通过适当的指针来由适当的位置检索代码字。该指针然后就被更新到该编码流中的下一代码字。

应指出，这一方法要求整个被编码的帧存贮在译码器，以及为了实际目的，存贮在编码器处。如果需要的是一个实时系统，亦即较少猝发数据流，可用两个帧缓存作为编码器和译码器的存储体。

要注意的是，一译码器是以一定的确定顺序对代码字进行译码的。采用并行编码，对编码流有确定的顺序要求。这样，如果能将并行编码流中的代码字在编码器处交错排列成正确的顺序的话，一单个的编码流就足够了。代码字以同样的顺序正合拍地传送至译码器。在编码器方面，一译码器的模型决定代码字顺序并将代码字组合成一单个的数据流。这一模型也可以是一实际的译码器。

在数据为可变长时，将数据位传送并行译码部件存在有一个问题。在分解一可变长代码数据流时需要采用位移位器来调整这些代码字。以硬件实现的位移位器经常是很昂贵而且/或者很缓慢的。对毕特移位操作器的控制决定于特定代码字的大小。这一控制反馈电路使得不可能快速执行可变长度位移操作。如果以一速度不足以与多个译码器相适应的单个位移位器来作分解数据流的操作，就无法发挥以单一数据流来供给多个译码器的优点。

本发明提出的解决办法是，将编码数据分配给并行编码器的问

题与译码中调整可变长代码字分开对待。 每一独立的代码流中的代码字被组装成为固定长度的字，称之为交错排列字。在信道的译码器端，这些交错处理字可利用快速电路连接数据线及一简单的控制电路来分配到并行译码器单元。

通常是使交错排列字的长度大于最大的代码字长，以使将在每一交错处理字中至少含有足以完成一代码定的毕特位。 这些交错排列字可以包含有可能的代码字和代码字的部分。图4说明一组示例并行编码流的交错排列过程。

按照译码器处的要求进行这些字的交错排列。 每一独立的译码器接收完整的交错排列字。 毕特移位操作就地在每一译码器中进行，维持关系的并行性。应看到在图4 中每一交错排列字的第一代码字是该组中保持为最低的一代码字。 例如第一交错排列字来自编码流1，以最低代码字(即# 1)起头。此后跟着的是编码流2 中的第一交错排列字，再后面是编码流3中的第一交错排列字。不过，下一最低编码字是No 7。因此在该数据流中的下一个字是编码流2的第二交错排列字。

采用实际的译码器来作数据流模型考虑了所有设计选择和延迟以生成交错处理数据流。 这对于具有译码器和编码器两者的双向系统来说，无论如何不是很大的代价。应指出，这可以通用于任何具有确定顺序的可变长(即不同大小)数据字的一并行组。

本发明可以应用现有的译码器，例如Q译码器或B译码器，作为并行多重使用的位流发生部件。 不过也可以采用其他编码和编码器。本发明采用的编码器连同它的相关的编码属于简单编码器。

在本发明中，利用具有简单编码而不是复杂编码(例如Q编码器

所用的算术编码或者B编码器所用的多状态编码）的位流发生器可获得很多优点。简单编码的好处在于硬件实现快得多而且简化许多，比复杂编码要求较小的集成电路构成。

本发明的另一优点可以改善编码效率。采用有限量状态信息的编码不能完全满足每一概率的香农(Shannon)熵边界。容许以一单个位流发生器来处理多概率或结构的编码要求降低编码效率的约束。去除这些多结构或概率类别所必须的约束就有可能采用那些接近满足香农熵边界的编码。

本发明现在这一优选实施例所采用的编码(及编码器）被称之为R编码。R编码为一包括行程长度代码(即R2(k)代码)在内的自适应代码。在本优选实施例中，R 编码经过参数化以便使许多不同的概率能为单一的译码器设计加以处理。而且，本发明的R 编码还能由简单的高速硬件进行译码。

在本发明中，R编码器利用R编码来执行编码或译码操作。在此优选实施例中，一R 编码器是由位流发生器和概率估算单元结合而成的组件。例如，以图1中，R编码器可能包含有概率估算组件102和位流发生器103的组合，及概率估算组件105与位流发生器106 的组合。

代码字表示最大或然码元(MPS)的运行。一MPS 代表具有大于50%的概率的二元判定的结果。另一方面，最小或然码元(LPS) 则代表具有低于50%概率的二元判定的结果。应指出的是，当两个结果具有相同的可能时，只要编码器和译码器作相同的选择，选定为MPS或LPS中哪一个就无关紧要。对于一给定的称为MAXRUN的参数，被压缩文件中的位序列被示于表1中。

表1　　位发生器编码

代码字	意　义
0	MAXRUN连贯的MPS
1N	跟随有LPS的N连贯的MPS N<MAXRUN

为编码，由一简单计数器对一运行中的MPS数加以计数。 如果计数等于MAXRUN计数值，就将0 代码字送进编码流中而将计数器复位。如果碰到一LPS，则将后面跟随N毕特位的1送进编码数据，此N特指LPS前的MPS码元数。（应指出，有许多方法指定N毕特位以描述行程长度）。再次将计数器复位。应看到N 所需要的毕特位数取决于MAXRUN的值。还可看到可以采用代码字的1的补数。

为译码，如果编码流中的第一毕特位为0，则将MAXRUN的值放进MPS计数器，并将LPS读数清零。然后消除0毕特位。 如果第一毕特位是1，则检验下面毕特位提取N毕特位，并将适当的计数（ N） 放进MPS计数器和设置LPS指示。然后消除包含1N 代码字的编码流毕特位。

R编码按表1中的规则产生 。要指出的是其中所给定的R编码$R_x(k)$的定义由MAXRUN规定。例如

MAXRUN: $Rx(k)=X * 2^{k-1}$,

这样

MAXRUN: $R_2(k)=2^{*}2^{k-1}$,

MAXRNU: $R_3(k)=3^{*}2^{k-1}$,

等等。

应指出的是R编码是Golomb编码的扩展。Golomb编码仅利用$R_2(.)$编码。本发明的R编码允许采用$R_2(k)$和$R_3(k)$两个编码，以及如希望的话还包括其他的$R_n(k)$编码。在一实施例中采用$R_2(k)$和$R_3(k)$编码。R_n中的n=2以及n等于任何奇数(如，R_2，R_3，R_5，R_7，R_9，R_{11}，R_{13}，R_{15})。在一实施例中的$R_2(k)$编码，其行程计数r被以N编码；此行程计数r被以K毕特位说明，因此1N就以K+1毕特来代表。而在一实施例中的$R_3(k)$编码，毕特位N可能包含有一毕特位指明$n<2^{(k-1)}$还是$n\geq 2^{(k-1)}$，以及是k-1或k毕特指明行程计数r，这样就使得变量N分别以总数k或K+1毕特来表示。在另一实施例中，在代码字中可以应用N的1的补数。在这种情况下，MPS就趋向于具有产生许多0产生的编码流，向LPS则趋向于产生具有许多1的编码流。

表2、3和4和5说明用于本发明的当前实施例的一些有效的R编码。应当指出，本发明中也可以采用其他行程编码。一替换的行程编码的$R_2(2)$的示例如表6中的所示。表7和8 说明本优选实施例中所用的编码示例。

表2 $R_2(0)$

P(0)=0.500	
未编码数据	代码字
0	0
1	1

表3 $R_2(1)$

P(0)=0.707	
未编码数据	代码字
00	0
01	10
1	11

表4 $R_3(1)$

P(0)=0.794	
未编码数据	代码字
000	0
001	100
01	101
1	11

表5 $R_2(2)$

P(0)=0.841	
未编码数据	代码字
0000	0
0001	100
001	101
01	110
1	111

表6 替换的$R_2(2)$

替换	$R_2(2)$
0000	0
0001	111
001	101
01	110
1	100

表7 优选的$R_3(2)$编码

优选的	$R_3(2)$
000000	0
000001	1000
00001	1010
0001	1001
001	1011
01	110
1	111

表8　优选的$R_2(2)$编码

优选的	$R_2(2)$
0000	0
0001	100
001	110
01	101
1	111

在这一优选实施例中，$R_2(0)$码不进行编码：输入0被编码为 0，输入1被编码为1，(或者相反)，理想的是概率等于50%。 在本优选实施例中不利用$R_3(0)$码。本优选实施例中的$R_2(1)$ 编码理想的概率等于0.707(即70.7%)，而$R_3(1)$理想的概率0.794(即79.4%)。$R_2(2)$编码理想的概率是0.841(84.1%)。下述表7说明接近理想的行程码，其中概率偏差由下式确定：

概率偏差$=-\log_2(LPS)$。

表 7

概率	概率偏差	最佳Golomb码
.500	1.00	$R_2(0)$
.707	1.77	$R_2(1)$
.841	2.65	$R_2(2)$
.917	3.59	$R_2(3)$
.958	4.56	$R_2(4)$
.979	5.54	$R_2(5)$
.989	6.54	$R_2(6)$
.995	7.53	$R_2(7)$
.997	8.53	$R_2(8)$
.999	9.53	$R_2(9)$

应指出的是该编码在下列情况中是接近理想的：如由概率偏差所指出的，此概率范围是相对均匀地覆盖空间，即使该理想概率在较高K值中并不如在较低K值那样区别开。

一固定K的$R_2(K)$称之为行程码。不过，一固定的K 对一固定概率仅仅是接近理想的。应看到当在一理想概率编码时，按照本发明的R码采用等频率的0和1N代码字。换句话说，一半时间本发明的 R编码器输出一编码，而在另一半时间此R译码器输出另一编码。 检验0和1N代码字的数量可以确定是否应用了最佳编码。这就是说，如果输出太多的1N代码字时，则行程就太长；另一方面，在输出太多的0代码字时，则行程就太短。

Langdon所采用的概率估算模型检验每一代码字的第一毕特位以确定源概率是高于还是低于当前的估算。根据这一判断，增加或减少K。例如，如果发现指明为MPS的代码字，此概率估算就是太低。因此，按照Langon，对每一0代码字将K加一。 如果碰到一指明低于后跟有LPS的MAXRUN MPS的代码字，则概率估算就太高。因而，按照Langdon，就对每一1N代码字将K增加1。

本发明使得可采用比简单地对每一代码字的K加1或减1更复杂的概率估算。本发明包括一确定要用的编码的概率估算组件状态。许多状态可利用同一编码。 利用状态表或状态机将编码分派给各状态。

在本发明中，概率估算改变每一代码字输出的状态。这样，在本发明中，概率估算组件根据代码字是以0还是1开始来增大或减小概率估算。例如说，如果输出-"0"代码字，就增加MPS概率的估算。另一方面，如果输出-"1"代码字，就减小MPS概率的估算。

现有技术的Langdon编码器仅只利用$R_2(K)$ 码并增加或减小每一代码字的K。而本发明则协同状态表和状态机利用$R_2(K)$和$R_3(k)$编码以使得能使用自适应速率。就是说，如果静态数据量小时自适应就必须较快来产生较理想的编码效果，而在静态数据量较大时，自适应时间就可以较长以便能选择编码来达到对其余数据部分的较佳的压缩。应当指出，当出现状态变化的可变数量时，特定的应用特点也可能影响自适应速率。由于R码的性质，对R码的估算很简单，需要的硬件很少但非常有效。

协同应用$R_3(K)$码使得能以较高分辨率覆盖更大的概率空间。图5中作出了按照本发明的概率估算状态表示例。如图5所示，概率估算状态表说明，一状态计数器和与表中各自独立状态相关连的编码。该表包含正的和负的状态。该表显示为具有37个正状态和 37个负状态以及零状态。负状态指明一与正状态不同的MPS。在一实施例中，当MPS为1时可采用负状态而在MPS为0时，可采用正状态，或反之。应指出的是图5所示的表仅只是举例，其他的表可能具有或多或少的状态及不同的状态配置。

起始，编码器为0状态，这是$R_2(0)$码(即无编码) 其概率估算等于0.50。每一代码字被处理过后，状态计数器按照代码字的第一毕特位增一或减一。在本实施例中，0代码字增加状态计数器的大小，而1状态的代码字减小状态计数器的大小。因此，每一代码字均使得状态计数器改变状态。换言之，概率估算组件改变状态。但是，同样的编码可能与连续的状态相关连。在这种情况下，完成概率估算无需改变每一代码字的编码。换句话说，对每一代码字状态是改变的；但是此状态在某一次被映射到相同的概率。例如，状态5至-5

全都采用$R_2(0)$码，而状态6～11与-6～-11则采用$R_2(1)$码。应用本发明的状态表，使概率估算能保持用同样的编码器以非线性方式进行。

应注意到，对于较低的概率同一的R码包括有较多的状态。这是因为在低概率时采用错误编码时效率损失很大。行程码状态表的特性是在每一代码字之后在状态间转移。在指定给因每一状态改变而改变编码的状态表中，当在低概率区状态转换时，编码在一非常接近熵效率极限的编码与远离熵效率极限的编码之间转换。这样，损失（以编码数据位数计）可能导致状态间的跃变。现有技术的概率估算组件，例如Langdon概率估算组件，因这种损失而降低性能。

在较高概率行程码中，因错误编码的损失没有这样大。因此，在本发明中在较低概率区加有附加的状态，从而增加了二正确状态之间的转换变化，由此来抑制编码的失效。

应当指出，在某些实施例中，编码器可能具有初始的概率估算状态。换句话说，编码器可以一予定的状态，例如状态18开始。在一实施例中，可采用不同的状态表，以使得一些状态被用于最先的少数码元以能加快自适应，而将一第二状态表用于其余的码元作较慢的自适应，从而能作概率估算的精确调节。在这种状态下，编码器有可能在编码过程中快速地利用较理想的编码。在另一实施例中，编码流可为每一结构指定一初始概率估算。在一实施例中，不是按照一固定的数（例如1）来递增和递减，而是代之所根据已被计数的数据量或数据变化（稳定性）量来使概率估算状态作可变数增大。

如果状态表是对称的，例如图5中的表那样，仅仅其一半（包含

零状态)需要被加以存贮或以硬件实现。在一实施例中，状态数以带符号的幅值(1)补数形式加以存贮以利用对称的优点。在这种状态下，可以取1 的补数的绝对值来利用该表确定状态并检验符号来确定MPS是1或是0。这就可能减少增量或减量状态所需的硬件，因为状态绝对值用来作表索引而1的补数绝对值计算很简单。另一实施例为增加硬件效率可用硬接线或可编程状态机代替状态表。 编码变换器用硬接线状态是此状态表的现在这一优选实现方案。

图6是说明本发明译码器硬件系统一个实施例的方框图。参照图6，该译码器系统600包括先进/ 先出(FIFO)缓冲器 601，译码器602，存贮器603，及结构模型604。译码器602包括多重译码器。 编码数据610耦合到FIFO 601以便为其接收。FIFO 610耦合来把编码数据送到译码器602。译码器602被连接到存贮器603，及结构换型604。结构换型604也被连接到存贮器603。结构换型604 的一个输出包含译码数据611。

在系统600中，输入到FIFO 601的编码数据610被排序并交错分配。FIFO包含排好顺序的数据。数据流被送到译码器602。译码器602要求以确定的顺序由这些数据流来求取数据。虽然译码器 602要求编码数据的顺序是非不重要的，它亦不是随机的。利用在编码器而不是在译码器按这个序列整理该编码字，使该编码数据交错成一单个的数据流。在另一个实施例中，编码数据610 可包括一单个的未交错的数据流(针对每一个结构槽的数据)，结构类别，或概率类别被附加到数据流上。在这种情况下，FIFO 610被存贮区610 所代替，以便在数据运送到译码器602前接收所有编码数据，以便适当地分割该数据。

由FIFO 601接收编码数据610，结构模型604决定当前结构槽。在一个实施例中，结构模型604根据先前象素和/ 或毕特位，决定当前结构槽。虽然在图未表示，但结构模型604 可以包括行缓冲器。行缓冲器提供必须的数据，或样板，通过它们，结构模型604 决定当前结构槽。例如，结构是以当前象素附近的象素值为基础，行缓冲器可以被用来存贮用于提供特定结构的附近的那些象素的象素值。

根据结构槽，译码器系统600从存贮器603取出译码器状态用于当前结构槽。在一个实施例中，译码器状态包括概率估算模块(PEM)状态及位发生器状态。PEM状态决定用于译出新编码字的代码。位发生器状态保留当前运行的位的记录。根据结构模型604提供的地址，该状态由存贮器603提供给译码602。该地址访问存贮对应结构槽信息的存贮器603中的一个位置。

一旦从存贮器603已经取出当前结构槽的译码器状态，系统 600就决定下一个未被压缩的位并处理译码器状态。如果需要，那么译码器602就译出新编码字和/ 或更新运行计数。如果需要，除了位发生状态外PEM状态也被更新。而后，译码器602把新的代译码器状态写入存贮器603。

图7是说明本发明的码器的一个实施例。参照图7，译码器包括移位逻辑701，位发生器逻辑702，"新K"逻辑703，PEM更新逻辑 704，新编码字逻辑705，PEM状态到高速存贮器逻辑706，编码一屏蔽逻辑707，编码-MaxPL，屏蔽及R_3分离扩展逻辑708，译码逻辑709，多路转换器710，及运行计数更新逻辑711。移位逻辑701 被连接来接收编码数据输入713及状态输入712(从存贮器来)。移位逻辑701的输出也被连接到位发生逻辑702，"新K"发生逻辑703及PEM更新逻辑704，

并作为他们的一个输入。位发生逻辑 702 也连接来接收状态输入712并产生译码数据输出到结构模型。新K逻辑703产生一个输出，该输出被送到编码-屏蔽逻辑707的输入。PEM更新逻辑704 也连接到状态输入712并产生状态输出(到存贮器)。状态输入712 也连接到新编码字逻辑705的输入及PEM状态-编码逻辑706。PEM状态- 编码逻辑706的输出连接到扩展逻辑708并由其接收。扩展逻辑708的输出连接到译码逻辑709及运行计数更新逻辑711。 译码逻辑的另一个输入连接到编码-屏蔽707的输出。译码逻辑709的输出连接到MUX710的一个输入。MUX710的另一个输入连接到状态输入 712 。MUX710的选择输入连接到新编码字逻辑705的输出。MUX710及扩展逻辑708的输出和编码-屏蔽逻辑707输出一起被耦合到运行计数更新逻辑711的两个输入。此运行计数更新逻辑711 的输出被包括在状态输出中送到存贮器。

移位逻辑701移入编码数据流的数据。根据编码数据输入及状态输入，位产生逻辑702产生译码数据给结构模型。新K逻辑703 也使用该被移入的数据及状态输入以便产生一新K值。在一个实施例中，新K逻辑703使用PEM状态及编码数据的第一位来产生新K值。根据新K值，编码-屏蔽逻辑707产生为下一个编码字的RLZ屏蔽。下一个编码字的RLZ屏蔽被送到译码逻辑709及运行计数更新逻辑711。

PEM更新逻辑704更新该PEM状态。在一个实施例中，使用现存的状态更新PEM状态。该被更新的状态被送到存贮器。新编码字逻辑705决定是否需要一个新的编码字。PEM状态-编码逻辑706使用状态输入712决定用于译码的编码。该编码被输入到扩展逻辑708，以便产生最大运行长度，现行屏蔽及R_3分离值。译码逻辑709译码

编码字以便产生运行计数输出。MUX710或者选择译码逻辑709的输出或者选择状态输入712，送到运行计数更新逻辑711。运行计数更新逻辑711更新该运行计数。

包括译码器700的本发明的译码系统600，以流水线方式工作。在一个实施例中，本发明的译码系统600判定结构槽，估算概率，译码编码字，及由运行计数产生信息位，以上全部均以流水线方式进行。该流水线译码过程可以与并联译码系统及编码系统结合使用，后者见包括在题目为"数据并行译码及编码的方法及装置"的转让给本发明共同受让人的目前未决专利申请[序号 08/ 016，035(1993.2.10)]申请中的描述。图8 叙述了译码系统的流水线结构的一个实施例。参照图8，用数字1-6编号的6 个阶段表示本发明流水线译码过程的一个实施例。

第一阶段中，判定现行结构槽(801)，第二阶段，在现行结构槽已经被判定以后，执行存贮器读(802) 将该结构槽的现行译码器状态由存贮器中取出。如上所述，译码器状态包括PEM 状态及信息位发生器状态。

在本发明的流水线译码过程的第三个阶段中，产生一个去压缩位(803)。这就使得结构模型可能使用一个信息位。在第三阶段期间，两个其他操作情况发生。PEM状态被转换成编码型式(804)及做出关于是否必须译出一新的编码字的决定(805)也发生在第三阶段。

在第四阶段中，译码系统处理编码字和/ 或更新运行计数(806)处理编码字及更新运行计数包括数个子操作。例如，译出一个编码字以便决定下一个运行计数或为现行编码字(806)更新运行计数。如果需要的话，当译码新编码字时，从输入FIFO 取出更多的编码数

据。在第四个阶段中发生的另一个子操作是PEM状态的更新(807)。最后，在译码流水线的第四阶段中，如果现行编码字的行程计数是0(808)的话，该新PEM 状态就被用来作出用于下一个编码的运行长度0编码字(以后说明)决定。

本发明的译码流水线的第五个阶段期间，带有更新PEM 状态的译码器状态被写入存贮器(809)并为下一个编码字(810) 产生移位。在第六个阶段中，完成下一个编码字的移位。

本发明的流水线译码，实际上是从关于是否开始译码过程的一个决定开始的。这个判断是根据在FIFO 中是否有足够的数据送到本发明的译码器而做出的。如果FIFO没有足够的数据，那么该译码系统被停止运转。在另一种情况下，当译码数据输出到一个不可能接收从译码器来的所有数据输出的外围设备时，也可认为译码系统可能停止运转。例如，当译码器正在给图象显示接口及其相应的图象电路提供输出时，而图象可能显示太慢，这时，就需要停止译码器工作，而让图象追上。

一旦作出译码过程开始的决定，由结构模型决定现在结构槽。在本发明中，通过检验早先的数据来确定现行结构槽。这种早先的数据可以被存入行缓冲器中并可以包括从现在行和/ 或早先行来的数据。图9是说明一二进制位图的一个实施例，这儿对一个给定信息位来自现在行及早先行的信息位被用作结构模板。 就一早先数据而论可使用一模板来设计行缓冲器的信息位，这样，就根据被检验的早先数据是否与该模板相匹配来选择现行数据的结构槽。这些行缓冲器可以包括位移位寄存器。 一个模板可以用来作每个n-位图象的位面。

在一个实施例中，当下一个流水线阶段期间通过输出一个地址给存贮器而选择结构槽。该地址可以包括一个予定数目例如三位的信息位，去辨认该位面。通过使用三个信息位，就可以辨认象素数据中的信息位位置。用来决定结构的模板也可以作为地址的一部分表示出来。把用来辨认位面的信息位及辨认模板的信息位联合起来以便在存贮器中产生一特定位置的地址，该存贮器包括由那些信息位所确定的结构槽的状态信息。例如，利用三个信息位决定一特定象素的信息位位置及在该模板中的每个早先象素中相同位置的十个早先信息位，这就可以产生13-位结构地址。

使用由结构模型产生的地址，去访问存贮器（例如RAM）就获得状态信息。该状态包括PEM状态及位发生状态。PEM 状态包括现行概率估算，而位发生状态保留现在运行的信息位记录。超过一个状态的地方使用同样的编码，PEM 状态可以不包括概率类别或编码名称，而是包括在一个表格内的索引，例如，该表如图5所示。当使用如图5所示的表格时，PEM状态将还提供最大或然码元（MPS）作为辨认该现行PEM状态是被定位在该表的正边还是负边的一种方法。位发生状态可以包括计数值及LPS是否存在的标记。在一个实施例中，还包括用于译出下一个编码字的当前运行的MPS值。在本明中，为了减小运行计数器所要求的空间，所以位发生器状态是被存贮在存贮器中。如果用于每个结构的计数器，在系统中所消耗的空间较小的话，那么位发生状态就没有必要存贮在存贮器中。

在一个实施例中，对每一个结构槽，存贮器包括如下的状态信息：

(1) 当前运行计数或运行长度零（RLZ）屏蔽；

(2) 如果在运行结束有一个LPS，用一信息位作标记；

(3) MPS的现行值；

(4) 包括MPS值的一个PEM状态，用于下一个编码字；

(5) 一个连续位。

正如以后将要说明的，连续位及RLZ 屏蔽可供增加位发生速度之用。在其他的执行过程中可以不需要连续位及RLZ屏蔽。连连续位及RLZ屏蔽将在下面更详细讨论。

用于一个结构槽状态信息的执行过程，并包括用来表示他们信息位的数目的两个例子如表8可所示。

表 8—每一个结构槽的位数

名　称	RS(12,7)位	RS(7,5)位
继续	1	1
LPS表示	1	1
MPS值	1	1
计数	12	7
PEM状态	7	5
总 数	22	15

在一个实施例中，当一个有效的运行计数是状态部分时，连续标记等于逻辑1，当在运行的末端输出LPS时或者当解码一个新编码字时，连续标记等于逻辑0。当在当前运行的末端有一个LPS时，LPS 现在的标记等于逻辑1。当连续标记是逻辑1时，记数指示当前运行计数当连续标记是0时，记数指示运行长度0编码字的RLZ屏蔽。MPS值表示当前运行的MPS是否是"0"或"1"，而PEM状态包括下一个编码字的MPS值。

应该注意，在流水线过程可以开始以前，可能要求该解码系统的一些初始化。这样的初始化可以包括解码器状态存贮器置零。这可能通过使结构模型借助于一个或更多的控制信号，顺序地产生每个存贮器位置的地址，同时位发生器及概率估算组件把零输入到存贮器，来实现。

在流水线的第三阶段中，在从存贮器（例如RAM）获得得译码器状态以后，如果没有全部译出一个编码，则输出一个信息位。在一个实施例中，本发明使用一位（标记）上面称之为连续位来指明结构槽当前运行计数非零，即就是运行计数比零大。根据连续位的状态，根据在运行的末端是否存在LPS，及当前编码字是否是一个运行长度零编码字，本发明产生一个信息位作为输出。如果连续位是真（例如是逻辑1）表示没有零运行计数，那么输出是MPS。否则，如果连续位不是真（例如是逻辑0），但是LPS是在运行末端（例如，逻辑1）那么，在运行的末端即发送出LPS。如果连续位不是真而LPS存在与否的指示也不是真，那么就对一个编码字检验编码数据流以便决定发送的特殊位。如果当前编码字不是RLZ编码字，那么发生的信息位是MPS。换句话说，如果当前编码字是RLZ编码字，那么，由本发明发生的信息位是LPS。

表9是由状态信息产生的信息位的真值表及RLZ 编码字检测的结果。在表9中"X"表示"不考虑"值。

表9 -位发生真值表

连续	LPS存在?	RLZ编码字?	输出
1	X	X	MPS
0	1	X	LPS
0	0	0	MPS
0	0	1	LPS

在本发明中，根据PEM状态，作出一个编码字是否是RLZ编码字的决定。PEM状态决定被使用的R-编码，不同的R-编码有不同的RLZ编码字。在一个实施例中，RLZ编码字由"1"信息位组成，然而，"1"的数目按照当前R-编码改变。例如，对R2(0)RLZ是"1"；对R2(1)RLZ是"11"等等，而对R2(12)RLZ是"1111111111111"。

当连续位被清零时，RLZ屏蔽被存入当前结构槽的计数区段，而不是存贮零的运行计数。RLZ屏蔽能使很快作出在编码数据中是否存在有足够的"1"信息位的决定，以便指定一个RLZ编码字。 因为-RLZ编码字至少由一个"1"信息位组成，RLZ可能是比最长的RLZ 编码字少一个信息位。在一个实施例中，对R2(0) RLZ 屏蔽是"000000000000"，对R2(1)及R3(1)RLZ 屏蔽是"000000000001"，对

R2(2)是"00000000011"等等，直到对R2(12)是"111111111111"。

为了实现表9的真值表，图10中叙述了位发生电路一个实施例。参照图10，位发生电路1000包括NOT门逻辑1001- 1002，OR 门逻辑1003-1004，AND门逻辑1005-1006及XOR门逻辑1007 。 NOT 门逻辑1002被连接接收其输入端的RLZ屏蔽信息。NOT门逻辑1002 的输出连接到OR门逻辑1003的一个输入。在一个实施例中，NOT 门逻辑逻辑1002的输出包含n信息位，这里n是一个整数等于最大的编码字数量减1。OR门逻辑1003的另一个输入连接以接收以第二个信息位开始的编码数据的n信息位。OR门逻辑1003的输出连接到AND 门逻辑1005的输入端。编码字的第一信息位也是连接到AND门逻辑1005。AND门逻辑1005的输出连接到OR门逻辑1004的一个输入端。编码数据的一个单独的信息位也连接到AND门逻辑1005的输入端中的一个。OR门逻辑1004的另一个输入连接以接收LPS是否存在的位指示。OR 门逻辑1004的输出连接到AND门逻辑1006的一个输入。AND 门逻辑1006的另一个输入连接到NOT门逻辑1001的输出。NOT门逻辑 1001 的输入连接到连续位。AND门逻辑1006 的输出连接到 XOR 门逻辑1007的一个输入。XOR门逻辑 1007 的另一个输入连接以接收当前MPS值。XOR门逻辑1007的输出代表流水线第三级所产生的信息位。

NOT门逻辑1001，OR门逻辑1004，AND门逻辑1006 及XOR 门逻辑1007执行表9中叙述的真值表。NOT门逻辑1002，OR门逻辑1003， 及AND门逻辑1005决定RLZ编码字是否存在。当连续位是逻辑1(例如，真的)时，那么输入到AND门逻辑1006是逻辑0，这样XOR 门逻辑1007的输入端中的一个是逻辑0。在这种情况下，MPS是由电路1000输出的，其值是不予考虑的。换句话说，如果连续位是逻辑0(例如，假的)，

那么，AND门逻辑1006的输出有赖于OR门逻辑1004的输出。 在这种情况下，如果LPS目前的指示是逻辑1(例如，真的)，在运行的末端就有一个LPS的指示，那么OR门逻辑1004 输出是逻辑1，这样AND 门逻辑1006输出一个逻辑1给XOR门逻辑1007，在这种情况下由XOR 逻辑1007产生的信息位包括与MPS相反的位，或LPS。

假如连续位是逻辑0而LPS目前的指示也是逻辑0，那么XOR门逻辑1007的输出是以RLZ编码字指示信号为根据。当位发生电路1000接收一组输入时，如果RLZ编码字是存在的话，NOT门逻辑1002及 OR门逻辑1003都产生"1"。AND门逻辑检测由OR门逻辑1003 输出的所有信息位是否是"1"，如果是，就确定"RLZ编码字"信号。当RLZ编码字信号被确定时，OR门逻辑1004输出一个逻辑1。 如果连续位是逻辑0(例如不是真的)，那么，AND门逻辑1006的输出是逻辑1。当 AND门逻辑1006的输出是逻辑1时，LPS是输出。然而，如果 AND 门逻辑1005不检测由OR门逻辑1003来的所有信息位是当作"1"，那么RLZ编码字信号是逻辑0，这样OR门逻辑1004的任何输出是低的而AND门逻辑1006的输出也是低的。在这种情况下，XOR门逻辑1007 的输出是MPS。

在一个实施例中，在从一个外存贮器读出的其他输入被锁存以前，该编码数据输入是稳定的。用来传输编码数据的编码数据总线的宽度是最长RLZ代码字的量。在一个实施例中，数据总线数量是13位宽。在这种情况下，如果RLZ代码字存在，NOT门逻辑1003及 OR门逻辑1003产生13个"1"位。这里采用13个2位的OR门，使OR门逻辑1003产生13个"1"位。AND门逻辑1005接收OR门逻辑1003 所有的输出并检测它的输入的13位是否都是"1"。使用一个13输入AND门，使

AND门逻辑1005检测OR门逻辑1003的输出。如果13位都是"1"，那么AND门逻辑1005就确定"RLZ编码字"信号。因为所有RLZ编码字的第一位总是"1"，带有相关NOT门及OR门的一个RLZ屏蔽位即被删去。

在第三个阶段，概率状态与位发生被并行处理。在本发明中，PEM把PEM状态转换成一个编码。在一个实施例中，这个转换是用组合逻辑完成的。在处理PEM状态中，本发明决定当前编码，以及该编码的屏蔽以便计算编码流。处理PEM状态以获得R-编码的K值及R值。在一个实施例中，用7个信息位代表PEM状态，并被转换成4位信息代表K值及1位信息表示该编码是否是R3编码。图11 说明这种转换逻辑。7个信息位中的6个信息位被逻辑1101接收作为产生R3及K值的量。另一个信息位是用来辨认MPS值的符号信息位。如果R3指示位是在第一个逻辑状态(例如，逻辑1)，那么该编码是R3编码。如果R3指示位是在第二个逻辑状态(例如，逻辑0)，那么该编码是R2编码。表10是在逻辑1101的一个实施例中，说明组合逻辑的真值表。

表10-编码转换状态真值表

状态	K	R3
0	0	0
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	1	0
7	1	0
8	1	0
9	1	0
10	1	0
11	1	0
12	1	1
13	1	1
14	1	1
15	2	0
16	2	1
17	3	0

状态	K	R3
18	3	1
19	4	0
20	4	1
21	5	0
22	5	1
23	6	0
24	6	1
25	7	0
26	7	1
27	8	0
28	8	1
29	9	0
30	9	1
31	10	0
32	10	1
33	11	0
34	11	1
35	12	0

在另一个实施例中，R3信息位值，其编码转换状态可以提前执行。当PEM译码期间，K值及当前表位置（它的例子）可以然后一齐被存入存贮器并被存取。

使用由R3及K值所表示的该编码，本发明产生最大运行长度，屏蔽及R3分离值。在一个实施例中，产生的最大运行长度，屏蔽及R3分离值，是与使用逻辑实现真值表（如表11所示）同时进行的。参照表11，"X"表示"不考虑"。

表11—把代码转换到译码所需信息的真值表

k[]	r3	最大运行长度[]	屏蔽[]	r3分离值
0	0	1	0	x
1	0	2	1	x
1	1	3	1	1
2	0	4	3	x
2	1	6	3	2
3	0	8	7	x
3	1	12	7	4
4	0	16	15	x
4	1	24	15	8
5	0	32	31	x
5	1	48	31	16
6	0	64	63	x
6	1	96	63	32
7	0	128	127	x
7	1	192	127	64
8	0	256	255	x
8	1	384	255	128
9	0	512	511	x
9	1	768	511	256
10	0	1024	1023	x
10	1	1536	1023	512
11	0	2048	2047	x
11	1	3072	2047	1024
12	0	4096	4095	x

表11中，屏蔽值等于$(2^K)-1$。这个屏蔽值也被用作当前RLZ屏蔽。如果R3标记是表示该R3码是用来作译码的，(例如R3 标记被设置为逻辑1)，则R3分离值等于$2^{(K-1)}$。最大运行长度早已被说明。

在流水线的第三阶段中，还并行发生的是本发明决定是否需要一个新编码字，当译码处理不是在运行的中间，就需要一个新编码字。换句话说，如译码器是在运行的中间，那译码处理就不需要新编码字。在一个实施例中，要作出是否需要一个新编码的决定是根据连续位及LPS标记。如果连续位是0，LPS也是0，那么就产生一个新编码字。图12 是完成新编码字决定逻辑的一个实施例的方框图。参照图12，电路1200包括OR门逻辑1201及NOT门逻辑1202。 OR门逻辑1201被连接用来接收连续位及LPS的目前标记 。OR 门逻辑1201的输出连接到NOT门逻辑1202的输入，如果连续位及LPS目前标记两者都是逻辑0那么NOT门逻辑1202 就输出所需要的新编码字标记。

还是在第三阶段中，产生一个K的新值。在一个实施例中，按照表12的真值表，根据编码数据的第一个信息位及PEM状态，产生该新—K的新值。

表12—K的新值真值表

PEM状态	编码数据的第一位	新K
0	0	0
1	0	0
2	0	0
3	0	0
4	0	0
5	0	1
6	0	1
7	0	1
8	0	1
9	0	1
10	0	1
11	0	1
12	0	1
13	0	1
14	0	2
15	0	2
16	0	3
17	0	3
18	0	4
19	0	4
20	0	5
21	0	5
22	0	6
23	0	6

PEM状态	编码数据的第一位	新K
24	0	7
25	0	7
26	0	8
27	0	8
28	0	9
29	0	9
30	0	10
31	0	10
32	0	11
33	0	11
34	0	12
35	0	12
0	1	0
1	1	0
2	1	0
3	1	0
4	1	0
5	1	0
6	1	0
7	1	1
8	1	1
9	1	1
10	1	1
11	1	1

PEM状态	编码数据的第一位	新K
12	1	1
13	1	1
14	1	1
15	1	1
16	1	2
17	1	2
18	1	3
19	1	3
20	1	4
21	1	4
22	1	5
23	1	5
24	1	6
25	1	6
26	1	7
27	1	7
28	1	8
29	1	8
30	1	9
31	1	9
32	1	10
33	1	10
34	1	11
35	1	11

在第四个阶段，译码一个新编码字包括多路转换最大运行长度或者该编码数据流的被屏蔽出来的一个计数。如果采用R3编码，在编码数据流中存入一个计数有三种可能的方法。 用逻辑把代码扩展成为最大运行长度及屏蔽信息。也要检测0的计数，以便使RLZ编码字屏蔽可以输出而不是运行计数输出。如果计数比1大，更新当前编码定的运行计数要求减小该计数。0的计数就要被RLZ 编码字屏蔽所取代。

作为由K值及R3值所表示的当前编码，被转换成最大运行长度，"1"编码字的屏蔽及第一个长编码字的R3编码的值，这里指的是R3分离值。 屏蔽值表示在一个编码字中用于一给定编码的有效信息位的数目。使用这个信息，本发明就能译码一个代码字。

首先判断在运行的末端，是否有一个LPS，去译码当前代码字。在一个实施例中，这个判断是通过检验第一个信息位去实现的。如果第一个信息位是逻辑1，那么LPS就在运行的末端。如果是这种情况，由代码字所代表的运行计数就被确定了。

通过一代码字表示的运行长度的确定是由译码逻辑1300 完成的，如图13所示。参照图13，译码逻辑包括NOT门逻辑1301， 移位逻辑1302-1303，AND门逻辑1304，加法器逻辑1305及多路转换器(MUX)1306 。 该最大运行长度连接到MUX1306 的输入并使其接收。MUX1306的另一个输入连接到加法器逻辑1305的输出，及AND门逻辑1304的输出，及下一个RLZ屏蔽(在第三流水阶段由编码—屏蔽逻辑产生)。AND门逻辑1304的输出连接到加法器逻辑1305 的输入端中的一个输入端。加法器逻辑1304的另一个输入连接到R3分离值。AND门逻辑1304的输入连接到移位逻辑1302 及1303。 移位逻辑

1303的输入连接到对于代码的屏蔽。移位逻辑1302 的输入连接到NOT门逻辑1301的输出，NOT门逻辑1301的输入连接编码数据。译码逻辑1300还包括产生LPS是否存在的指示的逻辑1310。

给定当前编码字，就决定了运行计数。对于"0"代码字，最大运行长度作为运行计数被输出。对于R2代码字，由NOT门逻辑1301 倒相的编码数据通过AND门逻辑1304与屏蔽值相"与"以便产生运行计数(这儿，移位器1302及1303不移位屏蔽)。对于短的R_3代码字（ 即就是，第一位和第二位是逻辑1的R_3代码字 ），该运行计数是通过NOT门逻辑1301把编码数据倒相，由移位器1302移位1，然后用AND门逻辑1304同由移位器1303移位1的屏蔽相与的结果。另外，对于 R3长代码字，如果第一位是1而下一位是0，那么编码数据由NOT门逻辑1301倒相，由移位器1302移位1并用AND门逻辑1304同屏蔽相与，然后由加法器逻辑1305加到R3分离值上。该R3分离值是第一长R3 代码字的运行计数。在本发明中，R3分离值是2的指数。 本发明的译码逻辑操作被概括在下面表13中：

表13—译码器操作

条件	运行计数输出
编码字0位是"0"	最大运行长度
代码字0位是"1"而代码种类是R2	屏蔽[11..0]AND(NOT代码字[12:1]
代码字0位是"1",代码字1位是"1"而代码种类是R3	屏蔽[11..1]AND(NOT编码字[12..2]
代码字0位是"1",代码字1位是"0",而代码种类是R3	R3分离+(屏蔽[10..0]AND(NOT代码字[12..2])

如果运行计数是0,那么就输出检验随后的一个RLZ代码字的屏蔽值,而不是运行计数。由译码器输出的LPS 现在位是简单的代码字位0。这被表示为逻辑1310,逻辑1310包括流水线延迟,以便使它产生与运行计数的产生相吻合。

在执行下一个操作过程中,产生一个新运行计数,图14 是更新下个运行计数逻辑的一个实施例的方框图。参照图 14, 更新逻辑1400包括减法器逻辑1401及多路转换器(MUX)1402 。 减法器逻辑1401被连接以便接收当前值并输出一个比当前值小的值。MUX被连

接以便接收当前值，减法器逻辑1401的输出，当前RLZ屏蔽及下一个RLZ屏蔽。MUX1402的输出是被更新的值，或下一个运行数。

在本发明中，该缺席情况是针对被更新的运行计数成为比当前运行计数小1。在这个情况中，MUX1402输出由减法器逻辑1401接收的值。运行计数0被RLZ代码字屏蔽所代替。当输出LPS时，RLZ 代码字屏蔽就无改变地通过了。这个操作是被概括在下面表14中：

表14—运行计数更新逻辑

条件	被更新值输出	连续位下一个值
当前值输入是比1大的运行计数(而连续位是"1")	运行计数-1	1
当前值输入是等于1的运行计数(连续位是"1")及被译码新代码字	下一个RLZ屏蔽	0
当前值输入是等于1的运行计数(连续位是"1")及新代码字未被译码	当前RLZ屏蔽	0
连续位是"0"	当前值(一个RLZ屏蔽)	0

还与第四阶段并行发生的是更新PEM状态。根据代码字的第一位，更新PEM状态包括增加或减少该状态。当当前状态是0，MPS是被转换而不是减少。在一实施例中，状态必须不超过一个最大值。

在本发明中，如果在状态0时，碰到"1"代码字时，就改变MPS；否则"1"代码字就使状态减量。而"0"代码字使状态增量，一直到最大状态为止。

PEM状态更新真值表如下面表15所示。

图15-PEM状态更新真值表

现在状态 0	代码字位0 0	下一个状态 0	转变MPS? 1
最大状态	0	最大状态	0
1至最大状态-1	0	现在状态+1	0
1至最大状态-1	1	现在状态-1	0

在一个实施例中，被更新的PEM状态与当前PEM状态相乘，以便使得仅在译码一个新代码字时，才进行更新。

还与第四阶段并行发生的是，使用表11采用同样逻辑把PEM 状态转换成一个代码并将其扩展为一个屏蔽来决定下一个RLZ代码字

(对于新PEM状态)，再根据表12中所产生的新K产生该代码字。在另一个实施例中，PEM状态更新逻辑(表15) 的输出连接到状态至编码转换逻辑(附加的逻辑与逻辑1101相同)的输入，以便产生新K值。

一旦第四个阶段被完成，该新位发生器状态及PEM 状态就被写入存贮器，应当指出，计数，连续位，LPS值，以及MPS值都能改变。在第五个阶段中，编码数据流被移位到下个代码字。在第6 个阶段完成移位操作。

图15是本发明移位器的一个实施例。参照图15，移位器1550包括寄存器1500-1503，滚桶形移位器1504，累加器1505及代码字大小测定单元1506。寄存器1500及1501被连接以接收编码数据。寄存器1500也被连接到寄存器1501。寄存器1501连接到寄存器1502，寄存器1502连接到寄存器1503及桶形移位器1504。寄存器1503 也连接到桶形移位器1504。累加器1505也连接到桶形移位器1504。代码字大小测定单元1506被连接以接收R3指示，K 值及从桶形移位器1504来的反馈，并输出一个信号送到累加器1505。在该优选实施例中，寄存器1501-1503是16位的寄存器，桶形移位器1504是32位到13位的桶形移位器，而累加器是4位的累加器。

寄存器1500-1501接收从FIFO送来的32位的字，而寄存器 1502-1503 32位的输出给桶形移位器1504。无论什么时候，至少有一个未译码的32位数据送给桶形移位器1540。用编码数据的两个32 位字予置四寄存器1500-1503使之初始化以便开始。该代码字大小测定单元决定"1"代码字是否存在，以及如果存在，那在"1" 以后的多少位是当前代码字的部分。在一个实施例中，代码字大小测定单元1506的操作如下面表16所示：

表16—代码字大小测定方框逻辑

代码字位0	代码字位1	R3?	输出
0	X	X	1
1	1	0	K+1
1	X	0	K+1
1	0	1	K+2

注意，K值是说明该编码的运行长度参数，即就是R2(K)或R3(K)，而"X"表示"不考虑"值。

累加器1505是一个四位的累加器用来控制桶形移位器1504，累加器1505通过从代码字大小测定单元1506 输入的值来增加它的值。当累加器1505溢出时(例如，每次移位计数是16 或更大)，寄存器1501-1503就被时钟启动移位。所有其他的16位移位使得从FIFO来的一个新32位字成为被请求的。累加器1505 的输入是代码字的大小，该大小由当前的代码字及当前代码字的最初1位或2位所决定。应该注意是某些实施例，寄存器1501-1504 在译码开始以前必须用编码数据使其初始化。

注意，本发明的作流水线式译码器，在启动结构模型之前不对每个结构槽的运行计数进行予解码。因为这样，存贮在存贮器中的状态信息包括一个无效运行计数状态。当对一个结构槽为无效运行计数状态发生时，就译码一个新编码字以便决定下一信息位。在初始化时，如果外部存贮器被置0，那么所有零的状态应该指示必须译码一个新代码字。所以在启动位产生以前，不必对每一个结构槽的运行计数进行予译码。如果结构槽需要予译码，所有的结构槽都必须至少有一个不管是被使用或未被使用的代码字。因为通常对一给定数据都有未被使用过的结构槽，这就影响压缩效率。即使压缩降低不显著，予译码亦使非实时软件编码器复杂化，并使实时硬件编码器实现极其困难。

片载存贮器可以用来存贮每个结构槽的状态信息。片载存贮器便于作较高速度操作及减少输入/输出(I/O)计数(减少整体费用)。在一个实施例中，如果希望的话片载存贮器只为较小的位发生器存贮足够的状态信息，而使用外部存贮器存贮附加的状态信息。存贮器的多个存贮分区可以被用来减少对一些存贮区使用内部存贮器及作为其他用途的外部存贮器所需要的I/O。片载存贮器还可以被用作高速存贮器以便存贮从共用的结构送来的信息。然后，信息可以送往译码系统供使用，而不是从外部存贮器中存放。在一些实施例中，片载存贮器只可用来存贮较长运行计数的最低有效位部分，及仅仅在运行的末端或当需要访问较高位进行计数时，才访问外部存贮器。

结构槽重复使用延迟是两个作用的组合。在结构模型决定当前结构槽以后，使用一个给定的结构槽以后及它能被再次使用以前需要一个延迟。而所需的另一个延迟则是在使用一组结构槽中的一

个结构槽以后及在能再次使用同一组中的另一个成员之前。

一个延迟是因为存贮器使用了多个存贮器分区以便增加带宽。例如，由于使用RAM的两个存贮分区，把结构槽分成两组，即使用RAM第一存贮分区的第一组，及使用RAM的第二存贮分区的第二组。对存贮分区进行访问，因此交替地使用两组结构槽。当一组存贮分区正在用于读取结构信息时，而另一组正在被写入。这就意味着结构再访问延迟是一个偶数的时钟周期。第二个作用是在能为一给定结构槽读RAM之前，必须经过足够的时间，因为它前面已被用于计算正确的状态信息（被更新PEM状态及位发生器状态），将其写到RAM及移位到下一个代码字。这些全部操作要求从流水线的第二阶段直到第六阶段来完成。

为了减小结构槽再访问延迟，有某些解决方法。在本发明中，一旦发生接着使用相同的结构槽模型，根据结构模型能指示解码器的概念，结构槽再访问延迟能够减少。此时，欲被写的结构槽数据可以被送出以便随后使用。这就是说，替代将数据写入而后再由存贮器读出，而是可将数据直接送出以兹利用。在这种情况下，就可避免对存贮器的过多的写和读操作。在另一实施例中，当同一结构槽被接着使用时，可采用一特殊的结构槽。这在一些情况下将降低压缩。不过，如果接着利用同一结构槽相当少见的话，这种压缩的降低将不会产生对性能的损坏。这些特殊结构槽可以是无需附加RAM的"无编码"结构槽。在另一实施例中，可将原始结构槽分成为"正常的"和"特殊的"结构槽。一单个结构槽可被用作针对许多正常结构槽的"特殊"槽。重访问结构槽的延迟也可依靠译码器中加速对第二和接着的请求处理来降低。换句话说，相继地处理同一结

构槽中的两个毕特位的时间，可能要少于独立地处理的时间。例如，如像状态至代码的变换操作，就比较容易。

本发明中采用的PEM不管可用的数据量如何均利用同一自适应方案。在另一实施例中，一种起始时较迅速自适应，而在更多可用数据出现时作较慢自适应的Bayesian 方法可被用来作更正确的估算。另外，还可将PEM固定在一现场可编程门阵列(FPGA) 或一可编程PEM状态表/ 机中。

结构模型是按应用确定的。因此，在一实施例中采用一外部结构模型。也可用可编程结构模型。

图16A是本发明的译码系统的一个实现方案的方框图。图 16A中，方框1601-1607为现场可编程门阵列(FPGA)。方框1601-1604由具有移位能力的位发生器组成，连接成用于由FIFO接口接收编码数据及由存贮器接收计数信息(例如位发生器状态) 。 位发生器块1601-1604还连接到方框1605由存贮器(经由块1605)接收PEM状态，以使块1605实施PEM。位发生器块1601-1604还连接到块1606以产生一计数和PEM存贮器地址。块1606还连接到块1607。块1607提供输出到接口电路，例如一视频接口，并可以连接到行缓冲存贮器。块1606及1607实现接口外还完成本发明的结构模型。这样 ，如图16A中所示，四个位发生器(块1601-1604)即共同使用由一FIFO给出的单一输入数据流来处理编码数据流，而PEM(块1605) 则为四个位发生器的每一个处理PEM状态。

所示的此四个位发生器作为同步的设备流水线共同工作。 图17说明此四个位发生器(例如块1701-1704)如何协同结构模型(例如块1706和1707)及概率估算组件(例如块1705)工作。为着重说明

此流水线结构，将对第一毕特位的操作绘出阴影。应当指出在当为进行下一次操作所要求的信息尚未就绪时位发生器设备具有空闲时间。应看到，多个位发生器并行操作是以错列时间开始的，以能使流水线的第三、第四、 第五和第六阶段的每一个为这些位发生器之一在任一时刻执行。换言之，此3-6 阶段中的每一个在每一周期执行。

依靠利用位发生器的错列安排，本发明不为置于第六阶段与第三阶段之间的反馈回路所限制。 每一位发生器在完成在第六阶段中移位到下一代码字之后，位发生器才能再在第三阶段开始。对于一单个位发生器，在当第三、第四、第五和第六阶段的任何一个的设备被利用时，其他阶段的设备没有适当信息使它们能运行。这样，位发生器的错列就使得此四个阶段的每一个并行地发生。

第三和第四阶段称为位发生器的代码字处理，而第五和第六阶段则称为位发生器的移位部分。在一实施例中，本发明采用代码字处理的时间多路化，而移位操作则被用来由一代码字处理部分和一移位部分生成两个位发生器。换句话说，用于一个位发生器的硬件作为两个虚拟位发生器动作。在这样一种结构中，两个位发生器和两个移位器可以时间多路方式被用以提供与四个图16A中的位发生器同样的译码。在这种情况下，数据仍然必须按照恰当的顺序，它应能保证对此二虚拟位发生器的每一个提供正确的数据。

图16B是这样系统的一实施例的方框图。在图16B中，FIFO接口被连接到移位器1611和1612。 移位器1611 和1612 还各自连接到BG1613和1614。BG1613和1614还通过块1605连接到PEM存贮器、由块1606和1607实现的结构模型、以及图16B中同样状态的计数存贮器。在这一实施例中，BG1613和1604无须移位。

编码过程需要相当数量的缓冲存贮器。考虑为什么一系统需要缓存。数据进入编码器的顺序与译码器的输出是相同的。但是由于许多熵编码器的局部位特性，代码字的顺序可能很不相同。多重编码器的任何一个均可能在送出一代码字之前编码数个毕特位。由于并行化，针对任一给定编码器的毕特位能由原始数据流中许多毕特位隔离开。这些插入位被送往其他的编码器，并可能产生数个，或者数百数千个代码字。不过，译码器首先需要代码字的第一毕特位。这样，对于这一方案来说将代码字重新排序是主要的。现提出一种对这些代码字作交错处理的方法。

数个编码流的交替处理引起的系统复杂性在于编码器上。当代码字被交错排列成数据流时，各个的编码流要被加以缓冲存贮。可能需要大量的存贮设备，特别是如果对于一结构槽的数据要比对于其他结构槽的数据偏移多得多，或者极少引入时。在这两种情况下，在该代码字中欲予编码的第一毕特位与最后位相距许多毕特位，而数个、可能是成百成千的其它代码字由插入的数据生成。这些代码字必须加以存贮。

对于一不均衡的系统，此时编码过程是在计算机（具有大量存贮器和/虚拟存贮器）上非实时地完成的，这就不成问题。但在实时VLSI实际方案中这种存贮器就需要很高成本。为了能以有限的存贮器进行编码，就需要在编码器与译码器之间存某些信号连系，或者在数据流中明显作出，或者是隐含的，此时达到存贮器的限制而且以适当方法作无损复元。这些方法对编码效率确实存在着不同的影响。

这种作明显的或隐含的信号连系可以由译码器之外的电路、甚

至作为编码器的部分来实现，如果该译码器具有一迫使一新代码字被加以译码的输入的话。 这一输入配合有足够的状态输出来使该外部电路与译码器同步，从而能够得到一将来能实时译码的" 连接结构"。

在数据流信号传输的一示例中，R2(K)的R3(K)码的定义被改变得包括其后不跟随有LPS的非最大MPS行程。 这是由加入不跟随有LPS的MPS的非最大长度行程来达到的。 这是以对最低概率时发生的代码字增加一毕特位来实现的。然后如果需要，就可以有用于非最大长度行程计数的可作唯一译码的前缀。表17说明-R2(2) 码及其特点。这种系统在未发生缓冲溢出时对编码效率影响很小，而在发生溢出时则有较大影响。

表 17

未编码数据	代码字	未编码数据	代码字
0000	0	000	111000
0001	100	00	111001
001	101	0	11101
01	110		
1	111		

以有限存贮器编码的一种方式是以隐含信号传输来模拟译码器上的缓存器。 为此采用的一方法是使编码器以有限再排序缓存器工作。可对每一并行编码器指派一缓存器。当开始一代码字时，对最终的代码字在恰当的缓存器中保留空间，并对每一代码字加以时间标记以确定代码字应被置入信道的顺序。 当一缓存器中的最后位置被保留给一新代码字时，一些代码字被放入压缩的位流而不管它们是否已被完全确定。

具有最低时间标记的部分代码字以选择一短的而且正确指明迄至目前所接收到的码元的代码字结束。例如，在一R 编码器系统中 ，假如必须在一具有 128 个最大行程的行程编码中为一系列100MPS提早完成一代码字的话，那就可以采用此针对128MPS的代码字，因为这正确地指明最先的100个码元。 在当此具有最低时间标记的代码字已经被完成时，就可将其移出再排序缓存器并加进编码流。这就可能使得先前完成的代码字也被置入编码流。 如果强制完成一部分代码字结果会使得一代码字由装满的缓存器移走的话，编码就能继续进行。如果一缓存器仍然是充满的，则该具有最低时间标记的代码字就必须再次加以完成并加进编码流。 这一过程继续进行直到原来装满的缓存器不再是满的。

图18中表示出了这样一编码系统。在图18 中，结构模型(CM)1801用来接收原始数据并将其送往编码器1802和1803。在一实施例中，CM1801将数据每隔一代码字送往编码器1802和1803。编码器1802和1803 对数据进行编码并将欲加缓存的数据分别送至缓存器1804和1805。缓存器1804和1805 每一个均包括有指明数据的临时顺序的时间标记。交错处理器1806由缓存器1804和1805 求取数

据以生成作交错排列的信道数据。每一缓存器1804和1805 向编码器1802和1803以及交错处理器1806提供一队列满指示信号，在数据由各自的编码器送往每一缓存器1804和1805 以及在数据进一步成为交错排列数据流的部分时作为请求机制指明缓存器1804 和1805的状态。

在这种隐含信号传输方法中的译码器保持跟踪与编码器中同样多的信息。译码器以时间标记和部分编码字维持再排序缓存器。在正常情况下译码器并不需要这些进行译码，仅仅用其来确定何时编码器因一缓存器装满而结束一部分代码字。 当译码器开始对一代码字进行译码时，它将一时间标记列入再排序队列。在译码器完成一特定代码字时，就将时间标记由再排序队列中去除。如果一缓存器装满，则具有最低时间标记的部分代码字就必须由此缓存器中移出。 任何没有由该代码字应用过的额外的结果因不对应于实时数据而被废弃。例如，译码器可能接收到一表明128MPS的代码字。在对100MPS译码后，译码器具有一装满的时间标记缓存器，并确定对应于编码器早期完成的128MPS的代码字。 该译码器忽略其余已被译码的28MPS，并在下次出现这一结构时利用一新代码字。 正像编码器那样，译码器继续舍弃代码字直到原先装满的缓存器不再是装满的。

图19说明一译码器系统实施例的方框图，图中，信道数据由缓存器1901接收，将其送至编码器1902和1904用于译码。编码器1902和1904对数据进行译码。经译码的数据为CM1906取得，再将其作为原始未加工的数据输出。时间标记缓存器1903和1905 指明它们的编码器的状态。例如，它们指出何时编码器装满数据。当出现这一

情况时，就产生一队列满指示到编码器，以指明此时编码器不再接收用于译码的任何更多的数据。当这种情况不复存在时，编码器就能重新接收要译码的数据。

作显明的信号传输的例子是一通报缓存器状态的独立的"状态"结构槽。这一数据被编码并送到编码流中一独立的译码器，可能但不一定是一专用的译码器。这一结构槽对每一交错处理的代码字均被作一次译码。这就是说，不管这是一个规则的（无缓存器溢出）交错处理代码字还是一个"强制的"（缓存器溢出）交错处理字。对一"强制的"响应是设计上的问题。一种可能的响应是把一毕特位信号当作为在数据流内作信号传输情况中的前缀的等同物。当该溢出位起作用时，将1n编码解释为无LPS值的MPS值运行。

如上所述，译码器将具有由数据流请求交错处理字的确定顺序。编码器必须对这一过程加以模拟来适应同一顺序。（注意，对于大多数可变长代码，编码器处产生的顺序将与译码器处消耗的顺序有很大的不同）。这种模拟随译码器单元的设计而定有可能十分复杂。模拟这一顺序的一种方法是完全重复编码单元处的译码器。这样，就能保证设计包括有在译码器处模拟中的任何给定设计中的存贮器和等待时间。

在本发明中编码器由独立的数据流产生代码字。这些代码字是毕特位组成的交错排列字。这些交错排列字由正在再生数据流中的探测译码器根据需要的基础上请求取得。每一被请求的交错排列字被送给该探测译码器，同时被置入信道中（或者信息缓存器内）。

这样一系统如图20中所示，图中输入数据为例如一结构模型或

状态机那样的序列机2001所接收。序列机2001 将数据依次分配到每一编码器2002-2004。每一编码器2002-2004 对接收的数据进行编码，然后分别存贮进缓存器2005-2007。交错处理机2008 用来接收由缓存器2005-2007所要求的数据。交错处理机2008根据探测译码器2009所提出的顺序由缓存器2005-2007请求并输出经编码的数据。缓存状况逻辑2010连接到交错处理机2008 指明缓存器2005-2007的状态，作为对译码器作隐含的或明显的信号传输中通告交错处理机2008的一种方式。 应看到此系统可以包括多于或少于三个编码器。代码字按要求传送给译码器，并对标记以与它的进行编码的同样的相继顺序加以译码。译码器系统如图21中所示，图中，被编码的信道数据为信道缓存器2101接收。信道缓存器2101 将数据送给编码器2101-2104以对它们进行译码。译码结果被送至序列机2105，例如一结构模型或状态机，由其输出数据。

本发明可被用于非二进制编码，例如Tunstall和霍夫曼码。

本发明的并行编码和译码可以采用非二进制编码， 例如 Tunstall 码。 Tunstall 码是可变长至固定长的代码。 换言之，Tunstall 码在编码过程中接收一可变长输入产生一固定长短的输出。因此，结果所得的编码数据流就包含同样长短的代码字。

Tunstall编码的一个实施例如表18中所示。应该指出的是表18中所示的码，除开该码是采用的长度3的字组外，均与未编码数据，或R2(0)码相同。

表18　　长度3二进制Tunstall码0.5MPS概率

输　入	输　出
mmm	111
mml	110
mlm	101
mll	100
lmm	011
lml	010
llm	001
lll	000

表　19

长度3二进制Tunstall码0.6MPS概率

输　入	输　出
mmmm	111
mmml	110
mml	101
mlm	100
mll	011
lmm	010
lml	001
ll	000

参照表18，"m"指示MPS，而"1"指示LPS。Tunstall码的其他例子示于表21之中。应注意，这一编码对具有接近0.60的概率时是很理想的，特别是对于概率在0.57与0.66之间时，比R2(0)码或R2(1) 码能更好地运行。这样，即使输出尺度较小，Tunstall 码在最初二R码间的间隙中亦能较R码更好地工作。较大输出Tunstall编码最终可以人为地到接近熵极限。

在本发明中，输出大小可以固定为任何位长度。由于是固定长度的输出，所有的数据排序操作就只需对这一尺度的代码字进行。例如输出大小可是8或12毕特。8位的代码字特别适用于软件运行。

因为代码字均为固定大小的，所以为每一译码功能块执行所需的仅只是简单的移位。 此移位逻辑可以每次位移同样数目的毕特位置而不必考虑数据流中的代码字，因为每一代码字的尺度相同。因此结合本发明采用Tunstall码，就使得进行译码期间所需的移位硬件和功能更为之简化。

另一系统示例是一JPEM霍夫曼编码的变型。(这一系统同样适宜于作MPEG霍夫曼编码)在这种情况下，硬件设备(即划分输入数据流)的优择以一序列发生器状态机实现。未被编码的标记依次传送到相邻接的编码器。例如，如果具有三个编码器，每一编码器就每隔三个标记接收一个标记。由于代码字是可变长度，连接成固定长度字以及再排序作信道交错处理的过程与前相同。 这样的系统如图20中所示。

这一系统的速度增加与编码器数成正比。 上述系统的速度将比一单个译码器几乎快三倍。这样，单一译码器每秒译码20×10^{6}标记时，三译码器系统则将以每秒60×10^{6}标记进行。 在另一实施例中，五个20×10^{6}标记/ 秒的译码器将以100×10^{6}标记/ 运行。

应指出的是，这已接近高分辨率电视(HDT)的最坏情况的标记速率。

这一系统还具有编码器所需的缓存很低而且是确定的(与数据无关)这种优点。因为霍夫曼编码为每一标记产生一代码字及标记在编码器之间相等地传送，所以可确定最坏情况缓存器长度。

考虑霍夫曼码具有一最小代码字长m，最大代码安长L，和一交错排列字尺度W。由于每一编码器每n个数据标记发送至少该最小m毕特位和最多该最大L毕特位(n为编码器数)，最坏的情况发生在当一编码器具有所有的最小代码字而另一个具有所有的最大代码字时。在一完全的交错排列的代码字产生之前的标记数为

$$\left(\left\lceil \frac{W-1}{m} \right\rceil + 1 \right) \text{。}$$

这样，对于此最坏情况所需的缓存为

$$\left(\left\lceil \frac{W-1}{m} \right\rceil + 1 \right) L \text{。}$$

一些典型的数可以是m=3，L=16，W=32。因而为每一编码器单元所需的缓存将为　　位。

$$\left(\left\lceil \frac{32-1}{m} \right\rceil + 1 \right) * 16 = 192$$

采用这种并行方法有两个可行的自适应霍夫曼编码实施例。一个例子中，根据的码元的一个字母的当前概率分布有数种可用的霍夫曼编码。此数据在大小上可能是任何毕特位数。 在代码字和数据值之间为1对1的映射。 这一类型的系统可利用码元的概率作为数据流分离器，即并行化功能。除这些差别外，本发明的交错处理完全如所描述地进行。

另一种可能是，霍夫曼码根据码元的自适应概率对哪一代码字代表哪一码元加以转换。这一代码可以与JPEG 基线霍夫曼编码同样的方式重复和实现。

非基线JPEG 图象压缩标准提出一系列表述量化变换系数的结构。这一结构模型在JPEG 静止图象数据压缩标准中有详细说明。虽然此标准是指明用于熵编码的QM编码器，但任一无损二元熵编码器均可能加以利用。而且，数种二元熵编码器可被用来处理不同的结构， 这些编码器也能与本发明的并行运行。特别是，数个QM 编码器或数个R编码器能被连同JPEG结构模型一齐加以利用来加速译码过程。

只要译码器是确定的，这种基本的交错模型就可能与任何编码配合工作。所以，例如说，QM编码器和R编码器就能并行采用。有损压缩器，例如JPEG可加应用，并且能与无损压缩器混合运行。 数据甚至可以是不同类型的。

在本发明的数据压缩系统中，一个值可以采用一个结构。例如，一特定的概率分布可以用霍夫曼码加以编码，如表20中所示。表20中对值-3至+3进行编码。

表20　　小整数的霍夫曼

值	估算概率	代码字
-3	0.0625	0000
-2	0.0625	0010
-1	0.25	01
0	0.25	11
1	0.25	10
2	0.0625	0011
3	0.0625	0001

如果表20中的编码被用作为位发生单元，就采用一种结构来对该值译码。

在本发明的一实施例中，对于不同类型的值可采用多重非二进制编码。在这种情况下，结构模型将由适当的非二进制结构取得下一个值，而不是如现有技术中那样取得数个二进制判定结果。

例如，如果要将一小整数在同一的-3至+3的范围内作二进制译

码，就必须利用多重结构。图22中列出了一典型的译码过程，其中结构被标号为1-6。结构模型将首先利用结构#1来确定该未知值是否等于零。在接到回答后，或者已了解该未知值，或者还需由它们自己的结构提出进一步的问题。这样，对一单个值进行译码就可能需要多重结构。对于较大的值，有可能需要针对正被译码的值中的每一毕特位需要对几乎二个结构加以译码。依靠采用单一的结构，本发明就避免了采用数个结构来对一个值进行译码。 而且依靠应用非二进制代码字，译码就可能快得多，结构模型也能简单得多。

应当看到的是，再排序电路的运行与采用二元判定相同，除了它是对代表整个值的代码字进行的。

像非基线JPEM图象压缩标准之类的系统依靠采用二元判定和与每一被压缩码元的相适应，能取得高性能压缩。不过，要将变换系数的幅值分解成数个二元判定的过程可能很慢。 为对一共有幅值为128的单一系数进行译码有可能必须多达16个二元判定。相反 ，霍夫曼码可能仅用一个操作来指明一个系数值。 霍夫曼码对于进行像一特定系数是否为零的判定进行编码将不是很有效的。 在数据交错处理系统中可能采用混合方法。一个QM 编码器可被用来确定数个二进制结果，然后一霍夫曼编码则可被用来指明系数的实际值。这两个编码的输出可以通常的方式作交错处理，霍夫曼码将使一些操作在速度上大大增加，而将QM编码器用于其他的操作则可保持高的压缩效率。

音频和视频交错处理在低位速率的压缩数字音频—视频系统中是一个问题。一种低位速率视频交错系统如图23中所示，图中 ，一音频编码器2301和一视频编码器2302 由各自的音频和视频源设

备产生编码数据。编码数据为需要者按照探测译码器2304 请求。此译码数据被输出到信道中作为单一的交错处理的信号流。 数据由信道缓存器2305接收，根据对数据的要求再将数据送往音频译码器2306或视频译码器2307。随着，音频译码器 2306 和视频译码器2307以再生格式的原数据输出。 音频和视频数据在同一信道上传送，必须加以同步地译码。通常是采用数据流内的标志来区别音频和视频数据。在现有技术中的这种标志器使得压缩效率降低。 采用本发明的交错处理，数据可根据音频和视频压缩器的要求加以交错处理，而无需标志器。交错处理使得能理想地压缩数据。

任何压缩系统的一个优点就是降低一组数据的存贮需求。 本发明的并行系统可以替代任何目前由无损编码系统所实现的应用。这些包括传真压缩、数据库压缩、位映象图形图象的压缩、 以及图形压缩标准例如JPEG和MPEG中变换系数的压缩。 本发明使得使对于不要求高速度的应用中亦能作小型高效的硬件和相对较快的软件实现的理想优择。

本发明胜过现有技术的真正优点在于以非常高的速度运行的可能性，特别是对于译码。以这种方式，本发明能促成对昂贵的高速信道的充分利用，例如高速计算机网络，卫星和地面广播信道。 图24描述了这样一个系统，其中，广播数据或者由一高速计算机网络提供数据到译码系统2401，后者对该数据进行译码产生输出数据。当前的硬件熵编码器（如Q编码器）将减低这些系统的工作效率。所有这些系统均被设计得以昂贵代价来实现高的带宽。 因译码器减低工作效率就会导致相反的作用。 本发明的并行系统不仅提供这些高带宽，而且由于数据能以压缩形式传送实际上更增加了有效带宽。

本发明的并行系统还能用来获得中速信道像ISDN、CD-ROM 及SCSI中更有效的带宽。这种带宽匹配系统如图25中所示，其中，来自例如CD-ROM，以太网络、小型计算机标准接口(SISI)或其他类似信号源的数据被送往译码系统2501，后者对数据进行译码生成输出数据。这些信道比一些当前的译码器速度更高。经常这些信道被用来协同要求比信道更高带宽的数据源工作，例如实时视频或以计算机为基础的多媒体。本发明的系统可实现带宽匹配的作用。

本发明的系统是对像新近出现的高分辨率电视(HDTV) 及MPEG视频标准那样的实时视频系统的熵编码器部分的最好优择。图26中作出了这样一个系统，其中，实时视频系统包含有与被压缩的图象数据耦合的译码系统2601。系统2601对数据加以译码，将输出送往有损译码器2602。有损译码器2602可以是-HDTV或MPEG译码器的转换、彩色变换和HDTV或MPEG译码器的二次采样部分。监视器2603可以是一电视或视频监视器。

尽管在阅读了上述说明之后熟知本技术领域的普通人员毫无疑问能对本发明可作出许多明显改变的和修改，但应清楚的是这里所介绍和描述的特定的实施例决不是用来作为一种限制。因此，在优选实施例中的详细情况并不能作为权利要求范围的限定，具有所提出的权利要求本身才是本发明的核心部分。

说 明 书 附 图

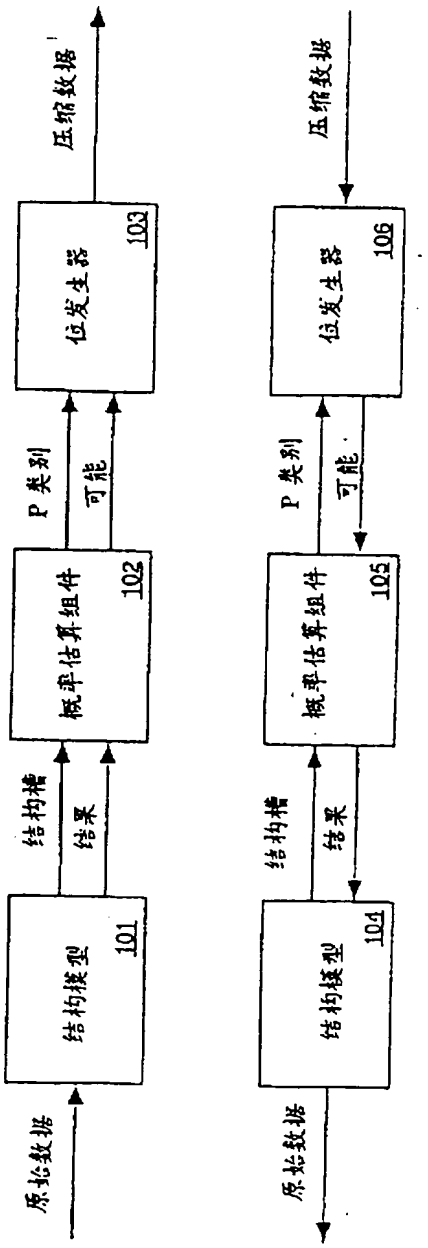

原始数据
结构模型
101
结构槽
结果
概率估算组件
102
P 类别
可能
位发生器
103
压缩数据
原始数据
结构模型
104
结构槽
结果
概率估算组件
105
P 类别
可能
位发生器
106
压缩数据

图 1

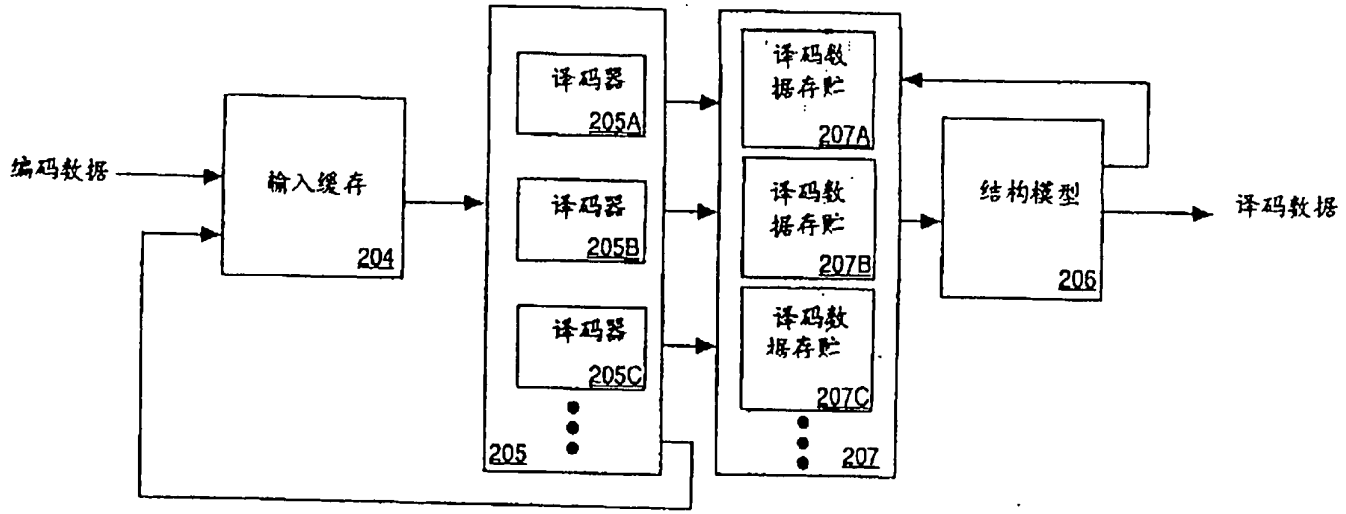

编码数据
输入缓存
204
译码器
205A
译码器
205B
译码器
205C
205
译码数据存贮
207A
译码数据存贮
207B
译码数据存贮
207C
207
结构模型
206
译码数据

图 2A

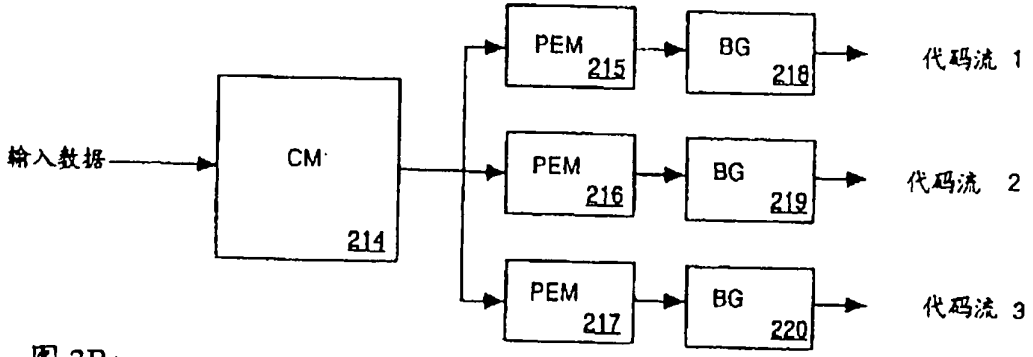

输入数据
CM
214
PEM
215
PEM
216
PEM
217
BG
218
BG
219
BG
220
代码流 1
代码流 2
代码流 3

图 2B

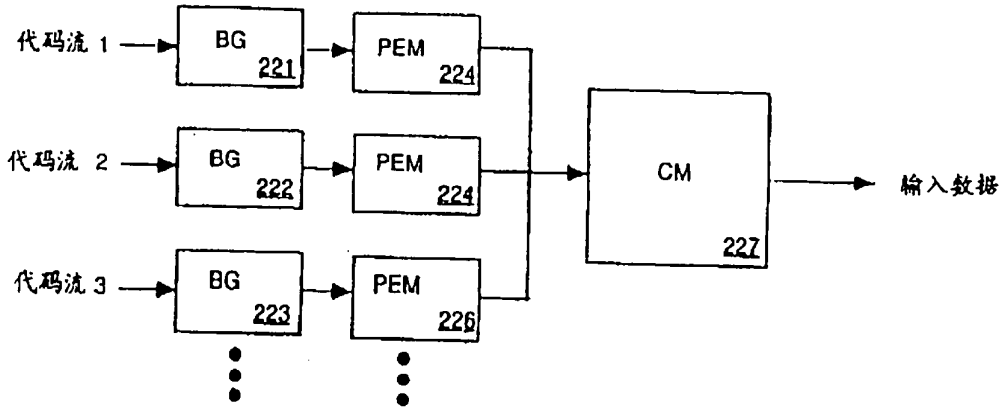

代码流 1
BG
221
PEM
224
代码流 2
BG
222
PEM
224
代码流 3
BG
223
PEM
226
CM
227
输入数据

图 2C

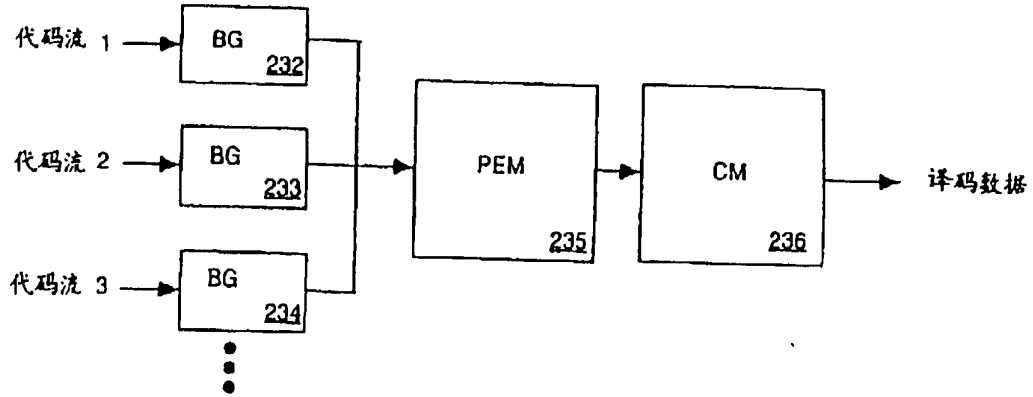

代码流 1
BG
232
代码流 2
BG
233
代码流 3
BG
234
PEM
235
CM
236
译码数据

图 2D

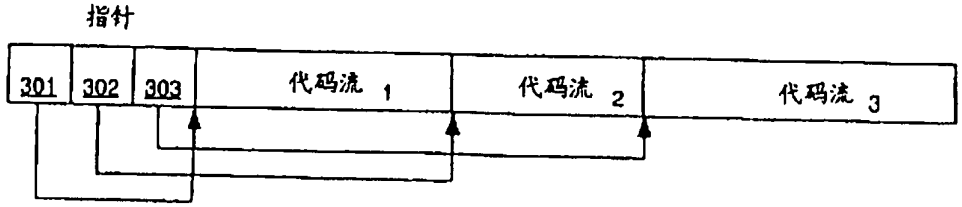

指针
301
302
303
代码流 1
代码流 2
代码流 3

图 3

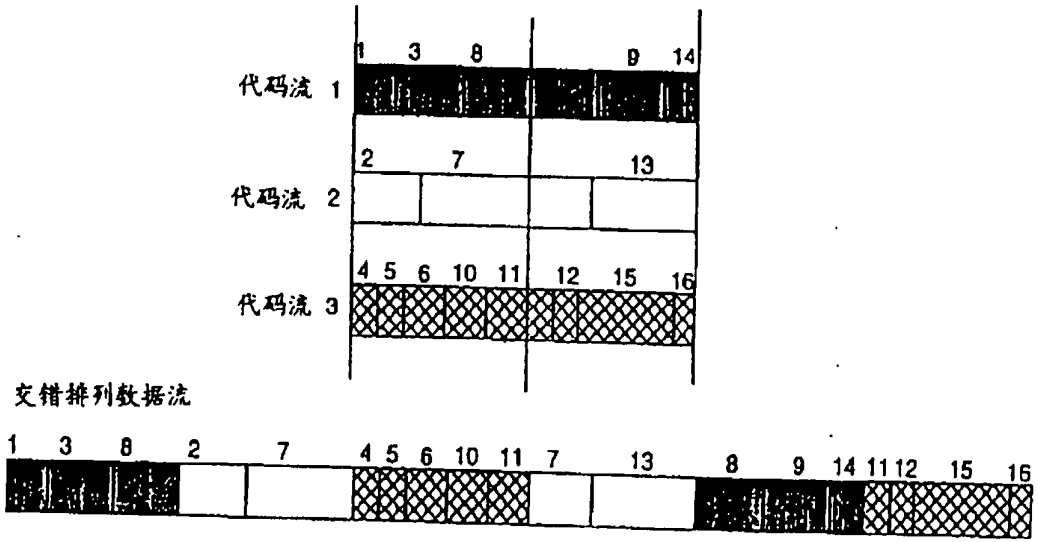

代码流 1
1 3 8 9 14
代码流 2
2 7 13
代码流 3
4 5 6 10 11 12 15 16
交错排列数据流
1 3 8 2 7 4 5 6 10 11 7 13 8 9 14 11 12 15 16

图 4

状态	代码
-35	R2(12)
-34	R3(11)
-33	R2(11)
-32	R3(10)
-31	R2(10)
-30	R3(9)
-29	R2(9)
-28	R3(8)
-27	R2(8)
-26	R3(7)
-25	R2(7)
-24	R3(6)
-23	R2(6)
-22	R3(5)
-21	R2(5)
-20	R3(4)
-19	R2(4)
-18	R3(3)
-17	R2(3)
-16	R3(2)
-15	R2(2)
-14	R3(1)
-13	R3(1)
-12	R3(1)

状态	代码
-11	R2(1)
-10	R2(1)
-9	R2(1)
-8	R2(1)
-7	R2(1)
-6	R2(1)
-5	R2(0)
-4	R2(0)
-3	R2(0)
-2	R2(0)
-1	R2(0)
-0	R2(0)
0	R2(0)
1	R2(0)
2	R2(0)
3	R2(0)
4	R2(0)
5	R2(0)
6	R2(1)
7	R2(1)
8	R2(1)
9	R2(1)
10	R2(1)
11	R2(1)

状态	代码
12	R3(1)
13	R2(1)
14	R3(1)
15	R2(2)
16	R3(2)
17	R2(3)
18	R3(3)
19	R2(4)
20	R3(4)
21	R2(5)
22	R3(5)
23	R2(6)
24	R3(6)
25	R2(7)
26	R3(7)
27	R2(8)
28	R3(8)
29	R2(9)
30	R3(9)
31	R2(10)
32	R3(10)
33	R2(11)
34	R3(11)
35	R2(12)

图 5

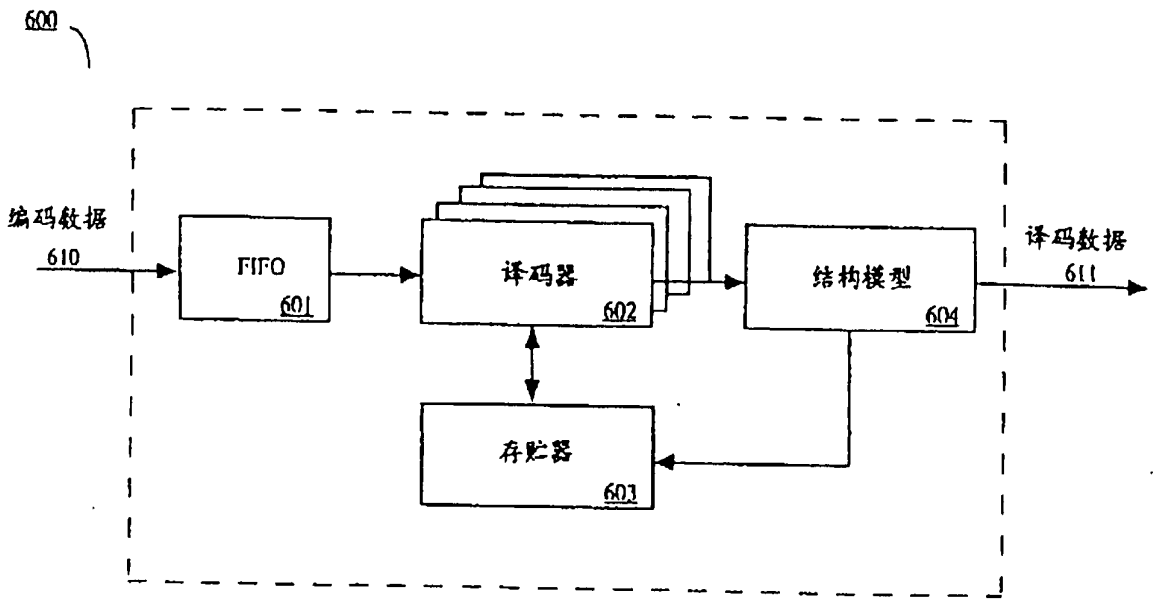
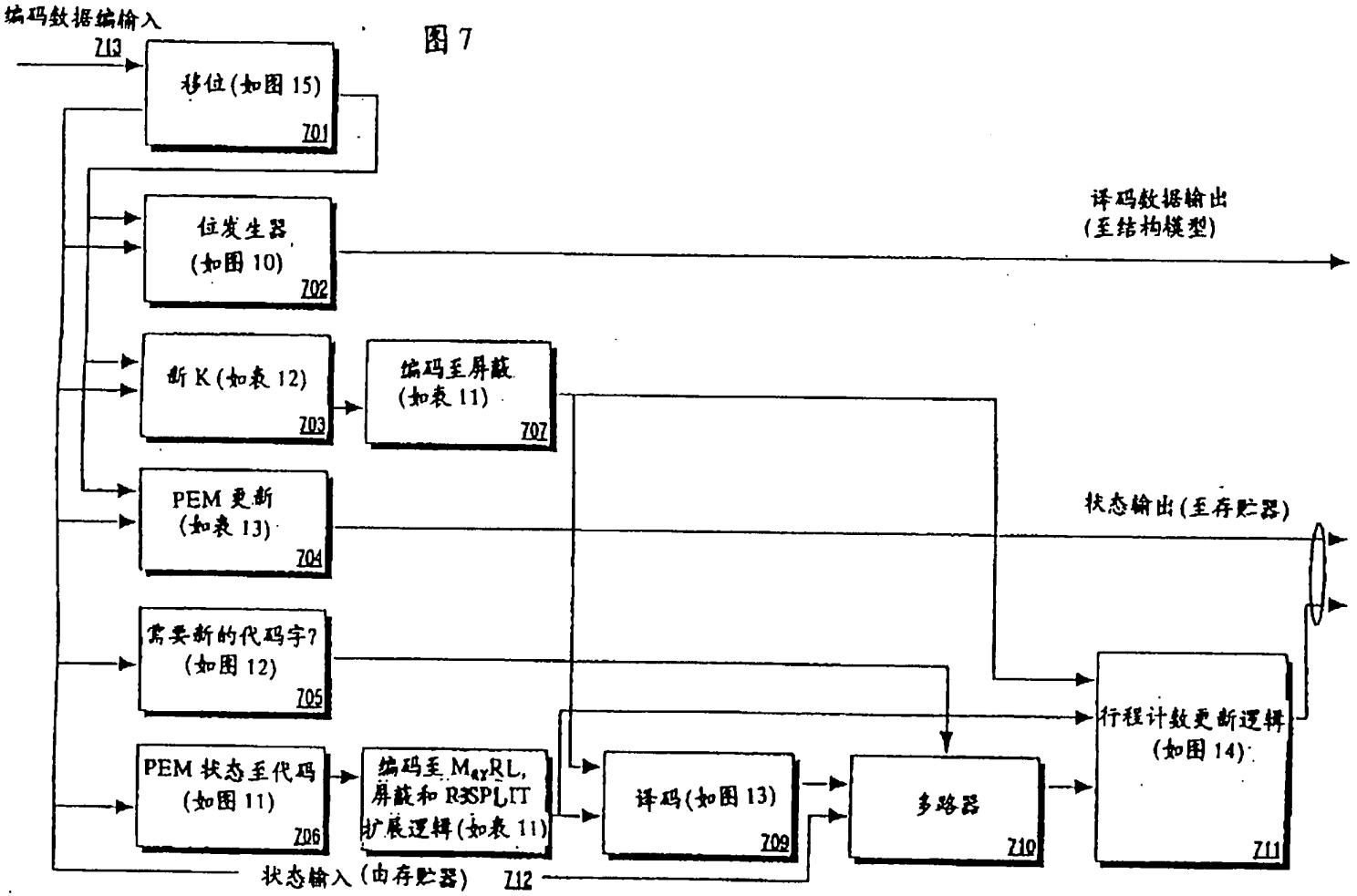


图 6

图 7


编码数据编输入
713
移位(如图 15)
701
位发生器
(如图 10)
702
译码数据输出
(至结构模型)
断 K(如表 12)
703
编码至屏蔽
(如表 11)
707
PEM 更新
(如表 13)
704
状态输出(至存贮器)
需要断的代码字?
(如图 12)
705
行程计数更新逻辑
(如图 14)
711
PEM 状态至代码
(如图 11)
706
编码至 MAXRL,
屏蔽和 R3SPLIT
扩展逻辑(如表 11)
译码(如图 13)
709
多路器
710
状态输入(由存贮器)
712

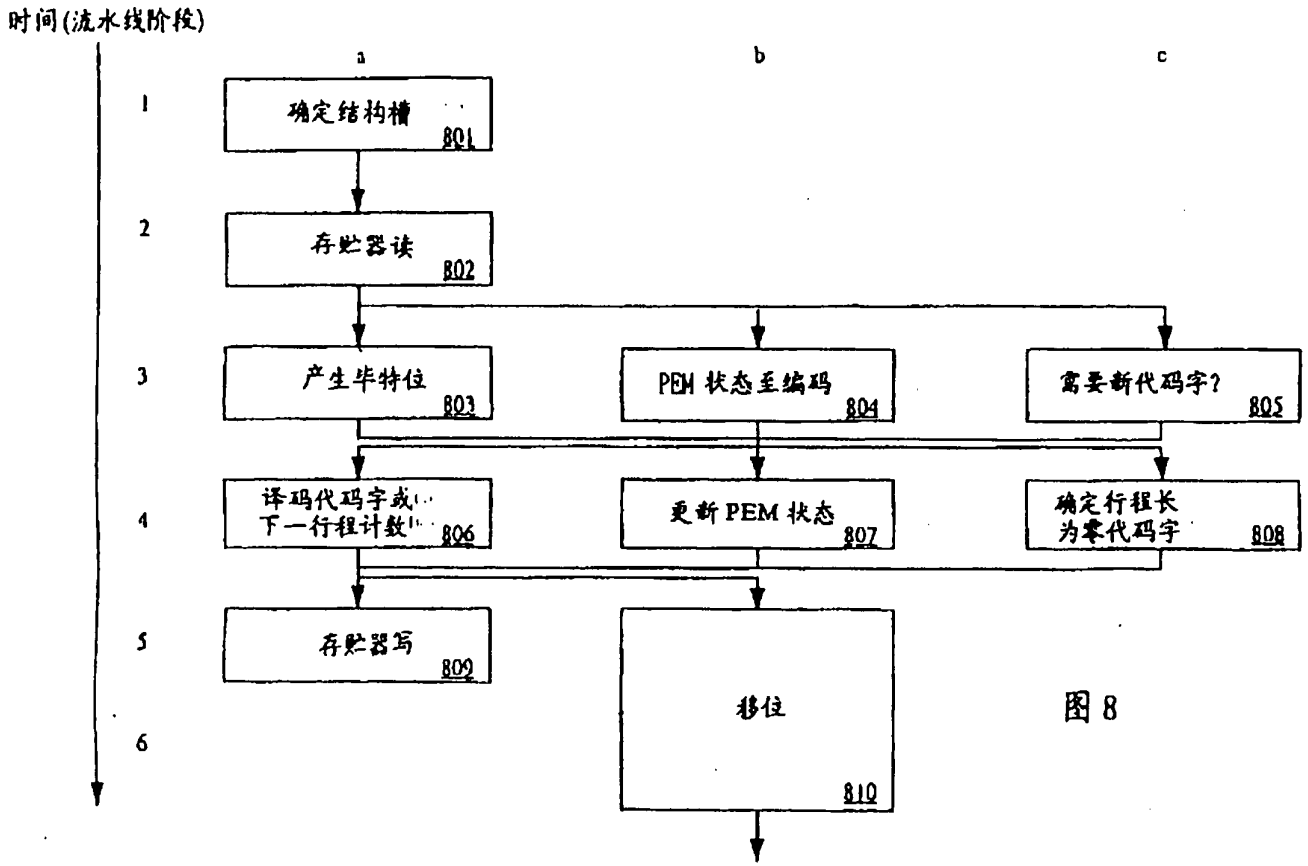

时间(流水线阶段)
a
b
c
1
2
3
4
5
6
确定结构槽
801
存贮器读
802
产生毕特位
803
PEM 状态至编码
804
需要新代码字?
805
译码代码字或
下一行程计数
806
更新 PEM 状态
807
确定行程长
为零代码字
808
存贮器写
809
移位
810

图 8

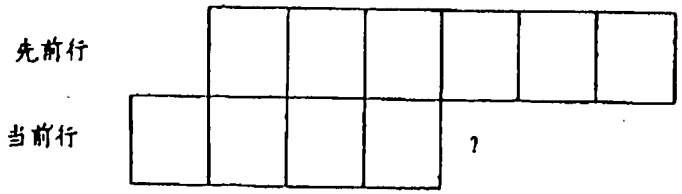


图 9

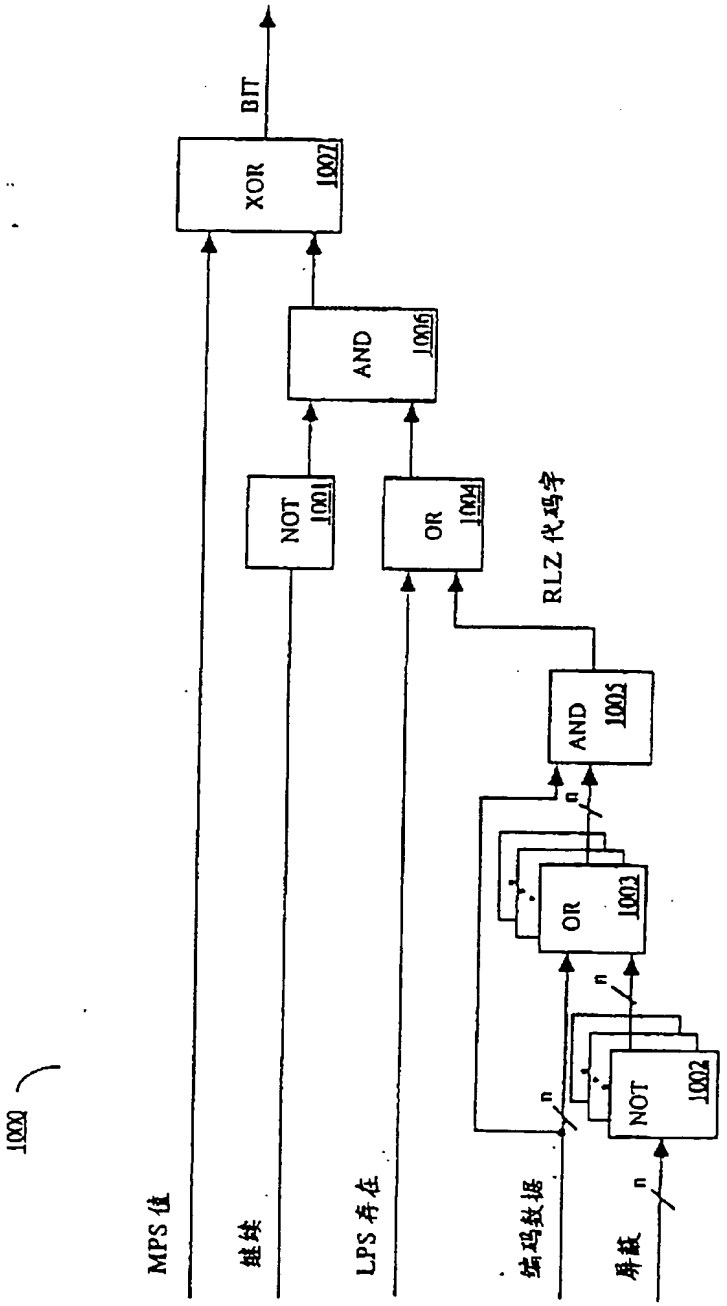

1000
MPS 值
继续
LPS 存在
编码数据
屏蔽
NOT 1002
OR 1003
AND 1005
RLZ 代码字
OR 1004
NOT 1001
AND 1006
XOR 1007
BIT
n

图 10

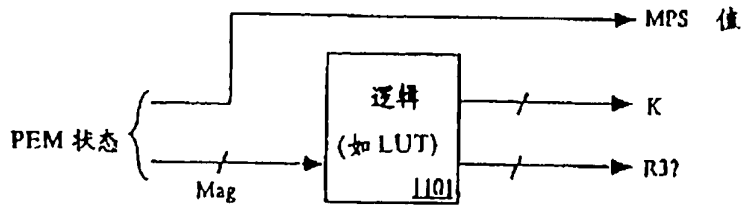


图 11

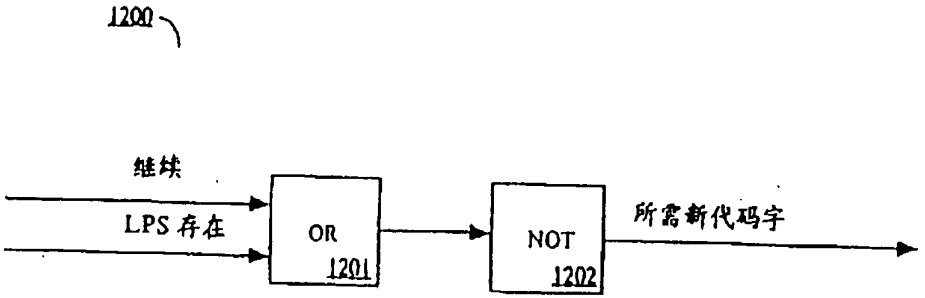


图 12

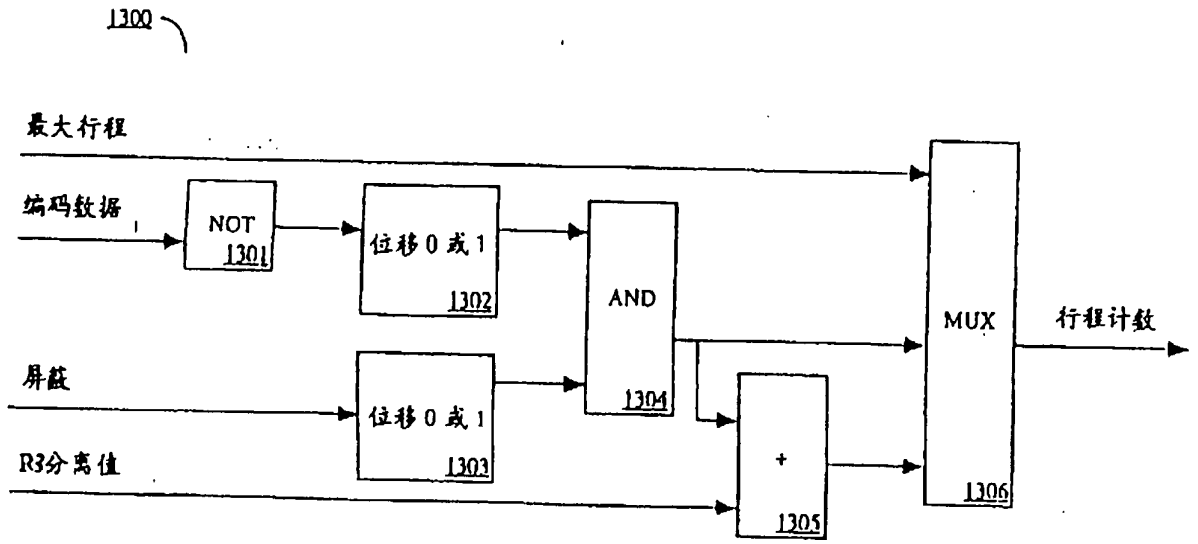

1300
最大行程
编码数据
NOT
1301
位移 0 或 1
1302
AND
1304
MUX
行程计数
屏蔽
位移 0 或 1
1303
R3分离值
+
1305
1306

图 13

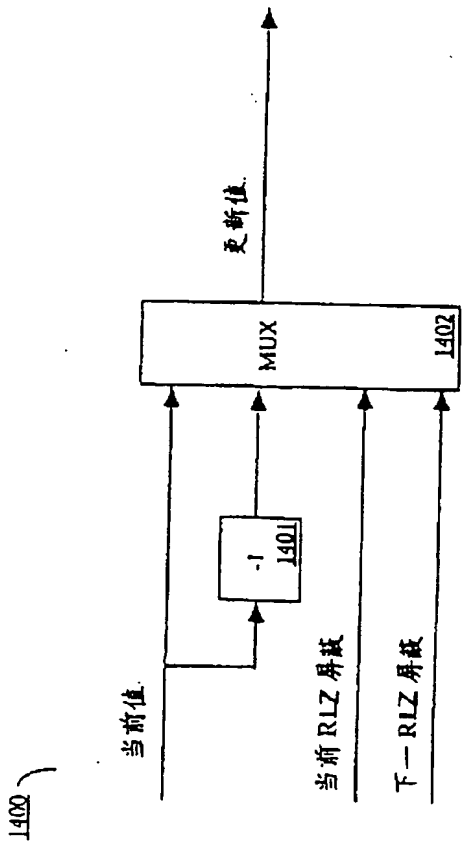


图 14

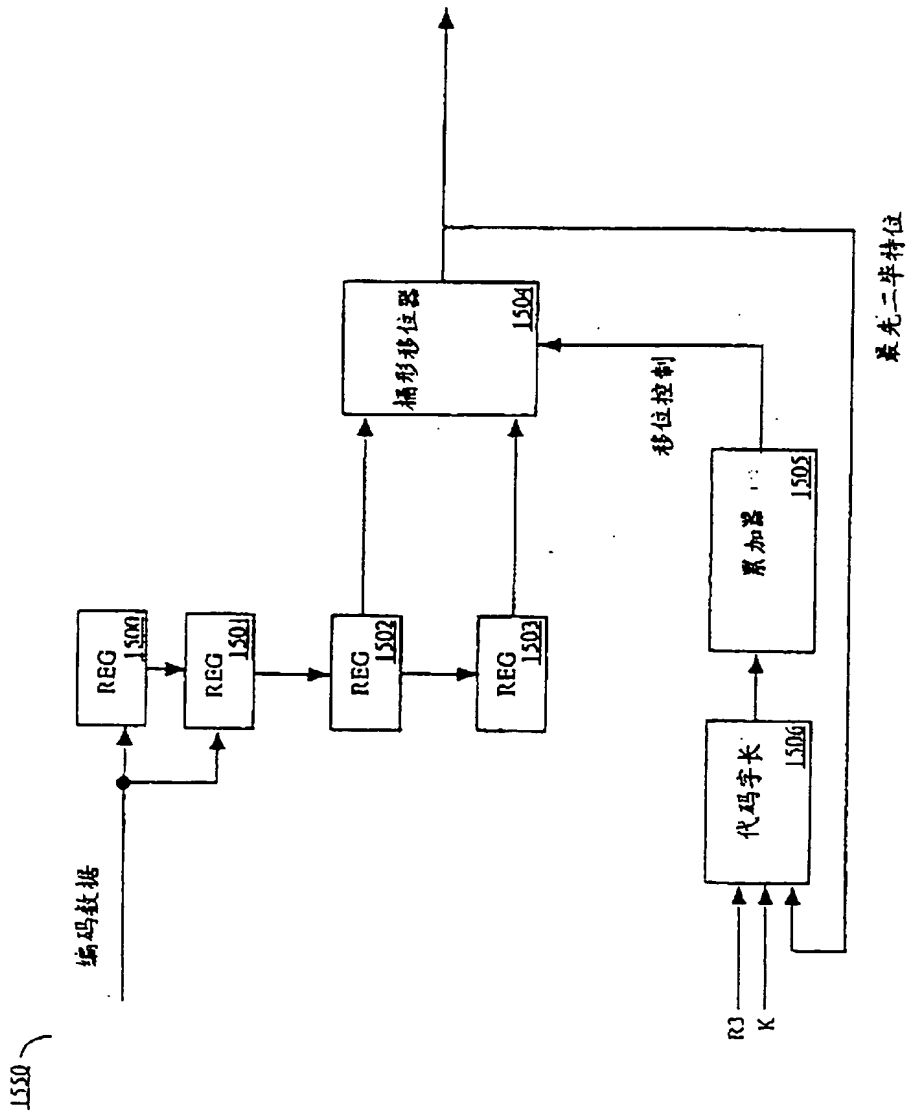

1550
编码数据
REG 1500
REG 1501
REG 1502
REG 1503
桶形移位器 1504
累加器 1505
代码字长 1506
移位控制
最先二进特位
R3
K

图 15

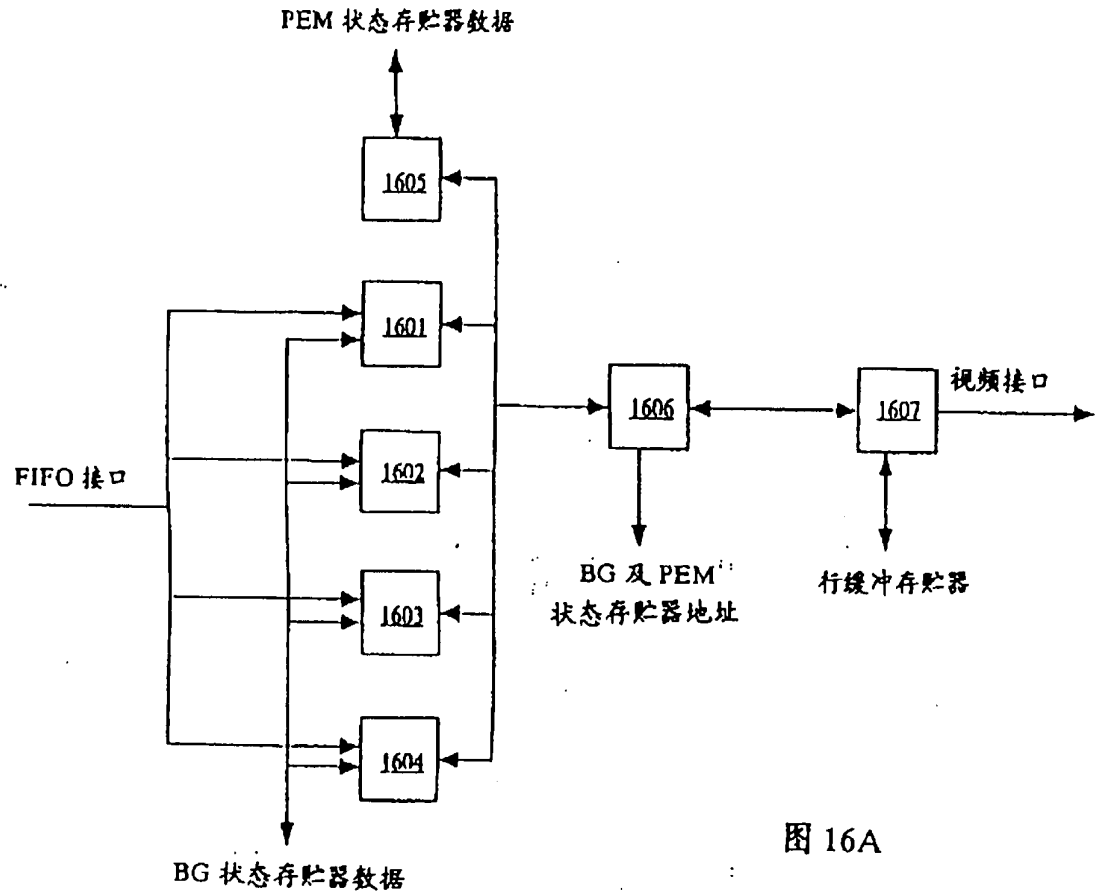

PEM 状态存贮器数据
1605
1601
1602
1603
1604
FIFO 接口
1606
1607
视频接口
BG 及 PEM
状态存贮器地址
行缓冲存贮器
BG 状态存贮器数据

图 16A

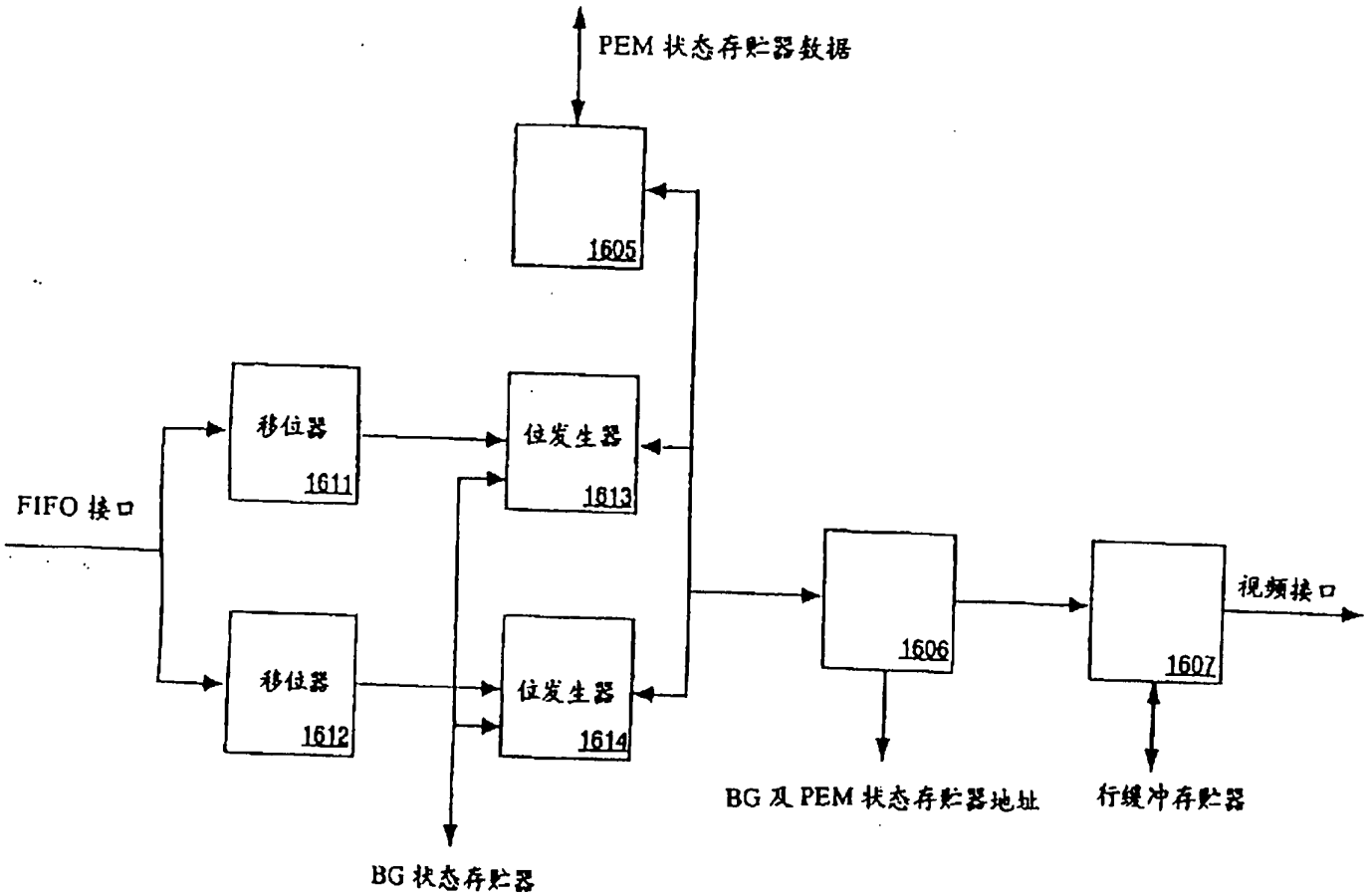

PEM 状态存贮器数据
1605
移位器
1611
位发生器
1613
FIFO 接口
移位器
1612
位发生器
1614
1606
1607
视频接口
BG 及 PEM 状态存贮器地址
行缓冲存贮器
BG 状态存贮器

图 16B

操作	被处理中的位(每位一时钟周期)											
CM: 确定结构槽		1	2	3	4	5	6	7	8	9	10	11
存贮器 A: 读				2		4		6		8		10
存贮器 B: 读			1		3		5		7		9	
BG (1/4): 产生毕特位							4				8	
BG (2/4): 产生毕特位				1				5				9
BG (3/4): 产生毕特位					2				6			
BG (4/4): 产生毕特位						3				7		
PEM: 状态变换为代码				1	2	3	4	5	6	7	8	9
PEM: PEM 状态更新					1	2	3	4	5	6	7	8
PEM: 新状态变换为代码					1	2	3	4	5	6	7	8
BG (1/4): 加计数 运行对代码字译码								4				8
BG (2/4): 加计数 运行对代码字译码					1				5			
BG (3/4): 加计数 运行对代码字译码						2				6		
BG (4/4): 加计数 运行对代码字译码							3				7	
存贮器 A: 写							2		4		6	
存贮器 B: 写						1		3		5		7
BG: (1/4) 桶形移位									4			
BG (2/4): 桶形移位						1				5		
BG (3/4): 桶形移位							2				6	
BG (3/4): 桶形移位								3				7

图 17

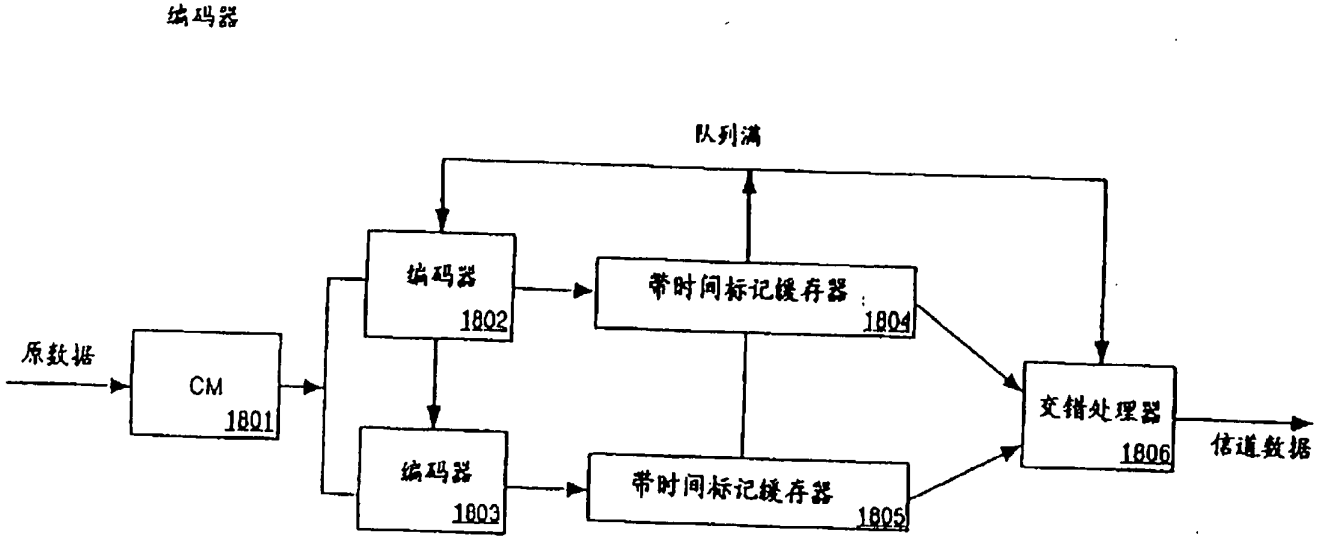

编码器
队列满
原数据
CM
1801
编码器
1802
编码器
1803
带时间标记缓存器
1804
带时间标记缓存器
1805
交错处理器
1806
信道数据

图 18

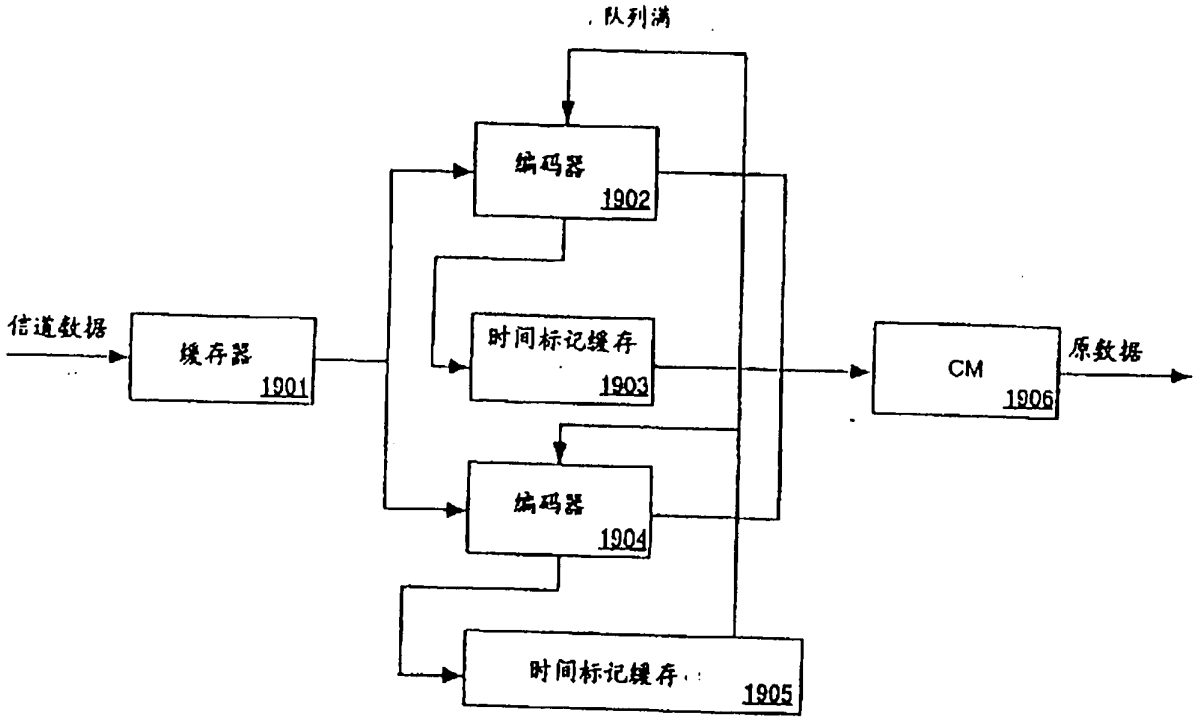

队列满
编码器
1902
信道数据
缓存器
1901
时间标记缓存
1903
CM
1906
原数据
编码器
1904
时间标记缓存
1905

图 19

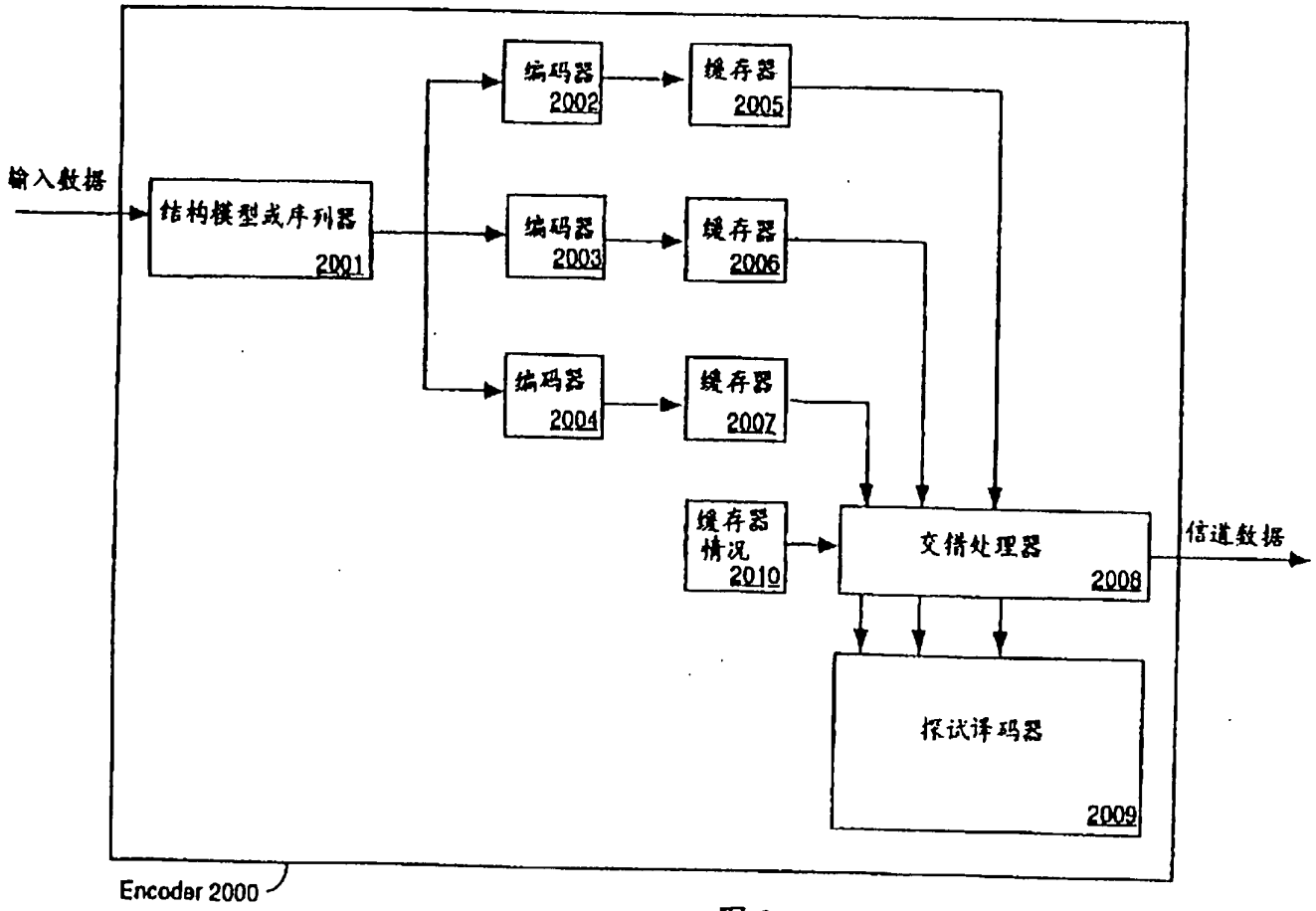

输入数据
结构模型或序列器
2001
编码器
2002
编码器
2003
编码器
2004
缓存器
2005
缓存器
2006
缓存器
2007
缓存器
情况
2010
交错处理器
2008
信道数据
探试译码器
2009
Encoder 2000

图 20

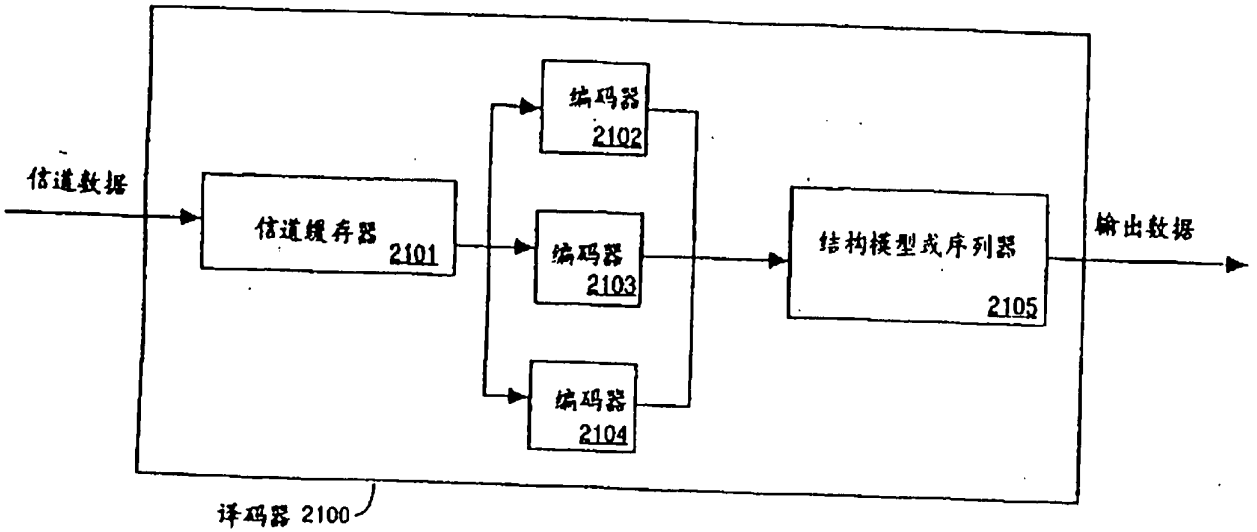

信道数据
信道缓存器
2101
编码器
2102
编码器
2103
编码器
2104
结构模型成序列器
2105
输出数据
译码器 2100

图 21

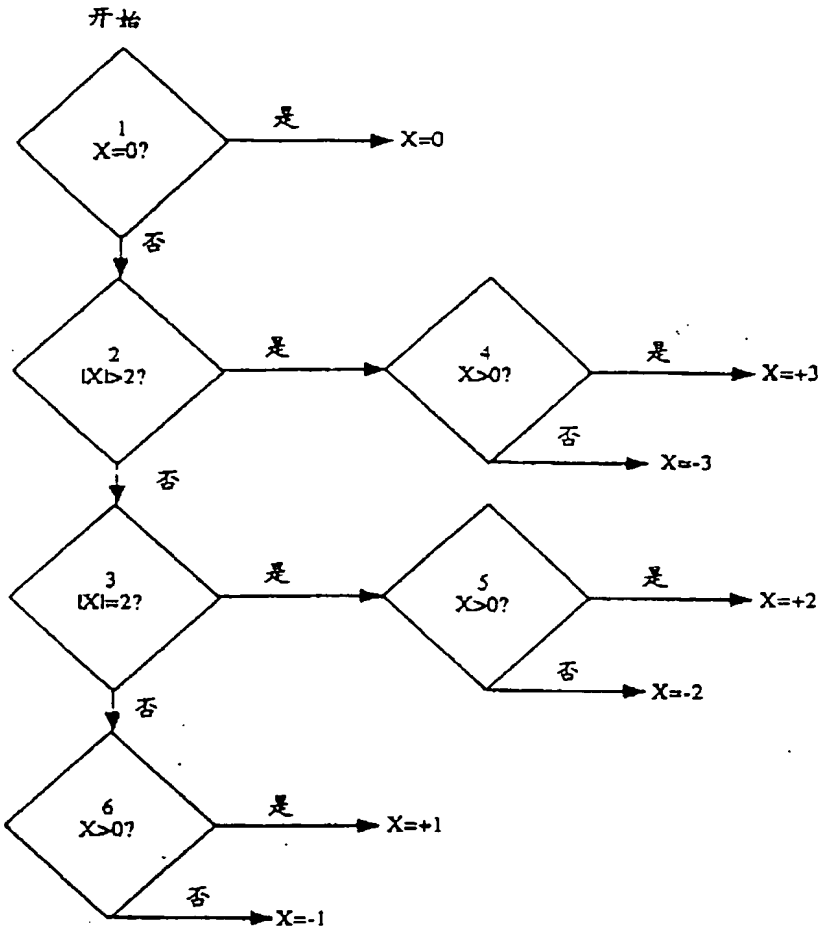


图 22

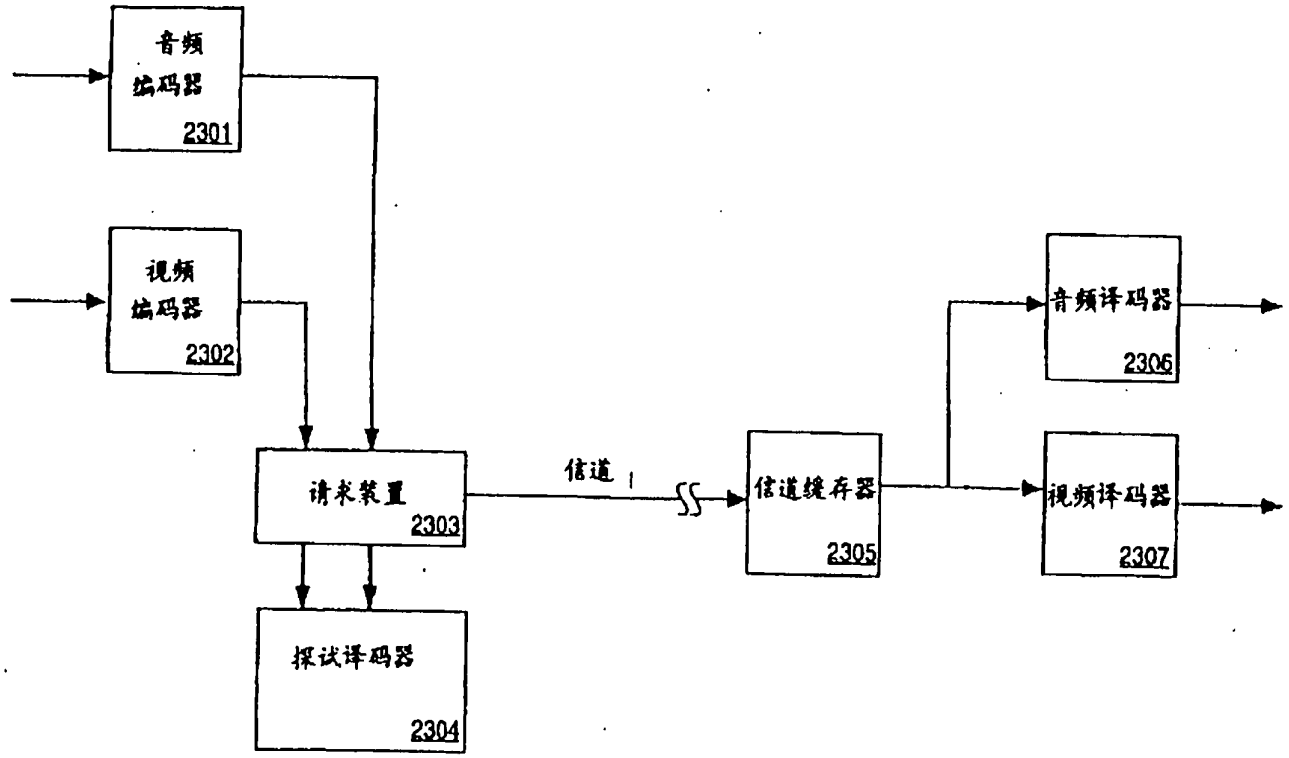

音频
编码器
2301
视频
编码器
2302
请求装置
2303
探试译码器
2304
信道
信道缓存器
2305
音频译码器
2306
视频译码器
2307

图 23

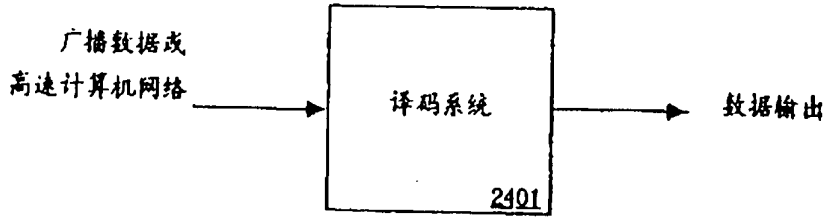

广播数据或
高速计算机网络
译码系统
2401
数据输出

图 24

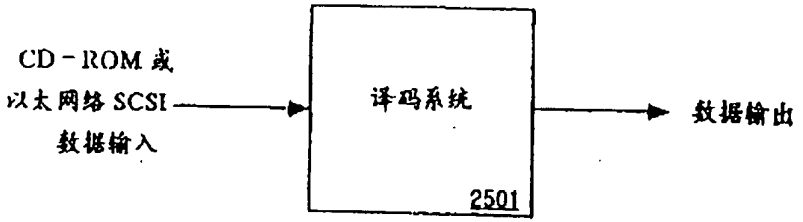

CD－ROM 或
以太网络 SCSI
数据输入
译码系统
2501
数据输出

图 25

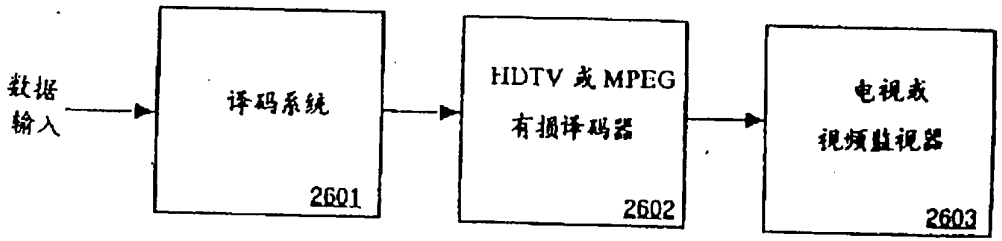


图 26